Escuela:

Escuela Superior de Ingenierías Industrial, Aeroespacial y Audiovisual de Tarrasa

Titulación:

Grado en Ingeniería en Vehículos Aeroespaciales

Alumno:

Javier Gallardo Alday

Título TFG:

Desarrollo de un sistema de comunicaciones en tiempo real, seguro, basado en la nube y orientado a drones.

Director del TFG:

Antonio Miguel López Martínez

Convocatoria de entrega del trabajo de fin de grado:

Enero de 2017

Contenido de este volumen:

MEMORIA

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Superior d'Enginyeries Industrial,
Aeroespacial i Audiovisual de Terrassa

2017

Desarrollo de un sistema de comunicaciones en tiempo real, seguro, basado en la nube y orientado a drones

ESCOLA SUPERIOR D'ENGINYERIES INDUSTRIAL,
AEROESPACIAL I AUDIOVISUAL DE TERRASSA
UNIVERSITAT POLITÈCNICA DE CATALUNYA (UPC)

Autor: **Javier Gallardo Alday**

Director: **Antonio Miguel López Martínez**

Enero de 2017

Resumen o *Abstract*

Español

Desarrollo e implementación de una plataforma digital basada en la nube que permite monitorizar, comunicar, y depurar distintos drones y dispositivos en tiempo real y de manera segura, aunque estén basados en distintas tecnologías y ubicados en distintos emplazamientos. Breve estudio de las comunicaciones inalámbricas empleadas por drones y dispositivos electrónicos durante sus operaciones habituales.

A lo largo de esta obra se presentan los requisitos del sistema, modo de trabajo habitual, procedimientos internos utilizados, detalles de la implementación y variedad de tecnologías que admite. En última instancia, se admite cualquier tipo de dispositivo con conexión a internet y que sea capaz de formar solicitudes http.

Se presentan asimismo soluciones complementarias de depuración de comunicaciones compatibles con la plataforma y un conjunto de herramientas y librerías desarrolladas para tecnologías y dispositivos específicos que permiten conectarlos al sistema.

English

Development and implementation of a digital framework based on the cloud which allows monitoring and debugging drones, and brief study of the wireless communications used by them during their normal operation. The solution provides a reliable and efficient way of achieving a real-time and secure data stream, allowing interaction between different communication technologies, regardless of their physical location.

This report covers the system requirements, common procedures, implementation details and supported technologies of the platform. The framework supports any type of device with an internet connection as long as it is able to form standard http requests.

Complementary solutions for communications debugging, always in the platform's format, are also presented. A set of tools and libraries developed for specific technologies and devices are also included, which significantly improve interoperability between all of them.

Prefacio

Lunes 1 de diciembre de 1783, Jardín de las Tullerías, no cabe un alfiler. Cientos de miles de personas se congregan en lo que se cree que fue la mayor aglomeración humana de la historia de París. El primer globo aerostático de la historia estaba a punto de realizar su viaje inaugural.

En la ciudad no se hablaba de otra cosa: la conquista de los cielos. Al menos eso creían. Cualquier estratega militar, por poca preparación o experiencia que tenga, sabe que la clave de una conquista no está en los efectivos, ni siquiera en los medios, sino en la comunicación. Y ahí, querido lector, es donde entra en juego la obra que tiene entre sus manos.

De vuelta al siglo XXI; quien estas líneas escribe cree que estamos ante otro intento de asaltar los cielos, esta vez mediante drones, de forma descoordinada: utilizando cada cual sus propias tecnologías.

Este proyecto, desarrollado como parte del trabajo de fin de grado, pretende proporcionar las herramientas adecuadas a aquellas personas que se dedican a desarrollar todo tipo de drones desde helicópteros hasta aviones pasando por multirrotores.

Se plantea el desarrollo de un sistema de comunicaciones para drones seguro, basado en la nube, y compatible con las tecnologías de comunicación más empleadas del mercado. El proyecto se desarrolla sobre tres ejes principales: la definición de los requisitos operacionales, el desarrollo e implementación del sistema, y su análisis y prototipado.

El ánimo de este trabajo es aunar esfuerzos y lograr establecer, como mínimo, un punto de partida sólido que en un futuro pueda dar lugar a un desarrollo en mayor profundidad de la solución que aquí se presenta para alcanzar mejores y más lejanas metas.

Le animo a proseguir con la lectura de esta obra teniendo presente este pensamiento de la aviadora más intrépida que ha conocido el universo: "*Nunca interrumpas a alguien que está haciendo lo que tú dijiste que no se podía hacer". (Amelia Earhart, 1936).*

El autor

Dedicado a mis padres, mis hermanos, abuelos y a otras muchas personas que tanto empeño han puesto en ayudarme a ser la persona que soy hoy en día.

1 SUMARIO DE CONTENIDOS

2 ÍNDICES DE FIGURAS Y TABLAS

2.1 ÍNDICE DE FIGURAS

2.2 ÍNDICE DE TABLAS

3 LISTADO DE TERMINOLOGÍA Y ABREVIATURAS

PHP: Hypertext Preprocessor[1]

OSI: Open Systems Interconnection

SQL: Structured Query Language

HTTP: Hypertext Transfer Protocol

HTML: Hypertext Markup Language

JS: JavaScript

IDE: Integrated Development Environment

SP: Service Pack

TCP: Transmission Control Protocol

UDP: User Datagram Protocol

XAMPP: Cross-Platform o XAMPP (X), Apache (A), MariaDB (M), PHP (P) and Perl (P)[2]

CSS: Cascade Style Sheet

[1] PHP es un acrónimo recursivo: que hace referencia a sí mismo.

[2] XAMPP puede ser interpretado o bien como acrónimo recursivo o bien como que la X hace referencia a "Cross-Platform" y es, entonces, un pseudo-acrónimo.

4 INTRODUCCIÓN

4.1 OBJETO

El objeto del presente proyecto es desarrollar una plataforma electrónica segura y basada en la web, accesible desde cualquier parte del mundo, para la monitorización y depuración de dispositivos inteligentes en tiempo real utilizando para ello tecnologías inalámbricas de conexión de bajo consumo y alta eficiencia.

Mediante esta plataforma debe ser posible, por ejemplo, consultar datos de cualquier dispositivo electrónico conectado a la red desde cualquier parte del mundo. Está orientada y especializada en el control de drones, pero es fácilmente generalizable a cualquier dispositivo electrónico utilizando librerías de conexión desarrolladas también.

4.2 ALCANCE

El alcance del proyecto es desarrollar todas las partes necesarias, que se detallan a continuación, para que la red digital DroneLink™ en tiempo real quede completa:

1. Desarrollo de una plataforma de monitorización de datos en tiempo real intuitiva, fácil de utilizar, y basada en internet utilizando consultas a base de datos SQL
    a. Desarrollo de la lógica de la aplicación del servidor en lenguaje PHP
    b. Desarrollo de la interfaz de la aplicación del servidor en lenguajes HTML y CSS
    c. Desarrollo de la lógica de la aplicación del cliente en lenguaje JavaScript
    d. Desarrollo de programas de ejemplo para la plataforma Arduino™
    e. Desarrollo de programas de depuración por puerto serie en Processing 3
    f. Escritura de la documentación de uso, instalación, e integración de la plataforma

2. Desarrollo de una estructura y tipo de datos SQL adecuada para esta aplicación
    a. Planificación de la base de datos (tablas, índices, columnas y flujos de datos)
    b. Implementación de la base de datos

3. Ensamblaje de una maqueta o prototipo para probar las funciones de la plataforma
    a. Desarrollo de las librerías necesarias de conexión en lenguaje C
    b. Desarrollo de programas de conexión mediante intermediario en lenguaje C#
    c. Ensamblaje de la maqueta

4. Desarrollo de librerías y programas que proporcionen conexión a la solución DroneLink™ en la mayor variedad de plataformas posibles y configuraciones distintas.
5. Estudio de soluciones más flexibles para la aplicación concreta a los drones como implementar la tecnología 3G/4G GSM

4.3 REQUISITOS

La plataforma digital DroneLink™ debe ser una plataforma:

- De adquisición de datos en tiempo real.
- Basada en internet y sus tecnologías.
- Accesible simplemente con un navegador.
- Que emplee, como mínimo, enlaces inalámbricos de bajo consumo XBEE.
- Que implemente otros enlaces de datos compatibles con la plataforma.
- Independiente del sistema operativo utilizado (compatible con Windows, MAC, Linux, y sistemas UNIX, entre otros).
- Compatible con los dispositivos móviles principales del mercado (iPhone® y Android®)[3].
- Que proporcione las medidas adecuadas de seguridad para evitar que la red se vea comprometida por ataques informáticos.
- Que sea personalizable y adaptable a cada aplicación práctica particular.

Es de especial interés remarcar que en este proyecto la seguridad ocupa un lugar tan relevante que es un requisito del mismo. Normalmente el aumento de la seguridad suele conllevar asociado pérdidas de rendimiento general. Se procurará disminuirlas al máximo; aunque en este caso prima la seguridad sobre el rendimiento.

4.4 JUSTIFICACIÓN DE LA UTILIDAD

Los dispositivos electrónicos que existen en el mundo son muchos y muy variados. Sin embargo, la necesidad de interconexión de estos dispositivos, o la de su monitorización remota, es también muy elevada. La mayoría de estos artilugios electrónicos necesitan trasladar información de un punto a otro: del dispositivo al usuario, de un dispositivo a otro, del usuario al dispositivo, etc. El problema surge cuando, para satisfacer estas necesidades de operación, se desarrollan soluciones concretas, limitadas, normalmente inseguras, de código propietario, y no estándares. Llegados a este punto es cuando surge la idea de crear una solución estándar, segura, accesible, rápida y universal para monitorizar estos dispositivos.

Por otro lado, tampoco son pocas las ocasiones en que en entornos de desarrollo como laboratorios o universidades se crea un dispositivo y, además de los esfuerzos que implica de por sí tal tarea, nos vemos obligados a programar un sinfín de herramientas de depuración para estos dispositivos. La plataforma DroneLink™ también está pensada para estos casos, permitiendo depurar estos dispositivos en un entorno altamente accesible, de actualización casi instantánea e interoperable.

[3] Todas las marcas son propiedad de sus respectivos dueños. Las marcas aquí (en la obra) mencionadas no guardan relación comercial ni de ningún otro tipo con el presente proyecto, su autor, ni su director.

La plataforma de comunicaciones DroneLink™ ha sido pensada y desarrollada específicamente para trabajar con drones. Esto no implica que no se pueda utilizar para otros dispositivos de diversa índole; pero sus herramientas están enfocadas al trabajo con drones.

La plataforma DroneLink™ es de gran utilidad no solamente conectando dispositivos distintos, sino que utilicen tecnologías distintas. La siguiente figura[4] (Figura 4.1) permite formarse una idea de las distintas tecnologías que relaciona la plataforma:

Figura 4.1: Tecnologías que implementa o relaciona la plataforma DroneLink™.

Fuente: Elaboración propia.

[4] Nota: Todas las figuras de este proyecto son de elaboración propia salvo indicación expresa. En caso de no ser de elaboración propia están protegidas bajo licencia Creative Commons (CC).

5 DESARROLLO

5.1 ANTECEDENTES O ESTADO DEL ARTE

Para abordar adecuadamente los antecedentes de un sistema como el desarrollado en este proyecto, por tratarse de algo novedoso, se ha considerado conveniente el estudio de pequeñas tecnologías y soluciones puntuales que se están aplicando hasta la fecha para abordar los problemas presentados en el apartado 4.4 [pág. 12]. También parece indicado dividir este apartado en dos grandes puntos: depuración de dispositivos e interconexión de los mismos. Se añade un punto al final con un enfoque específico a los drones.

5.1.1 Depuración de dispositivos

La depuración más común de dispositivos se lleva a cabo por el denominado puerto serie. Este puerto consiste en una interfaz de comunicación de doble sentido entre máquinas a través de la cual se canaliza la información deseada a una velocidad de transmisión específica medida en baudios (bits por segundo). Existe una segunda clase de depuración que se realiza monitorizando el prototipo en cuestión directamente con voltímetros, amperímetros y demás instrumentos de análisis.

Fuente: Pixnio.com (Licencia Creative Commons).

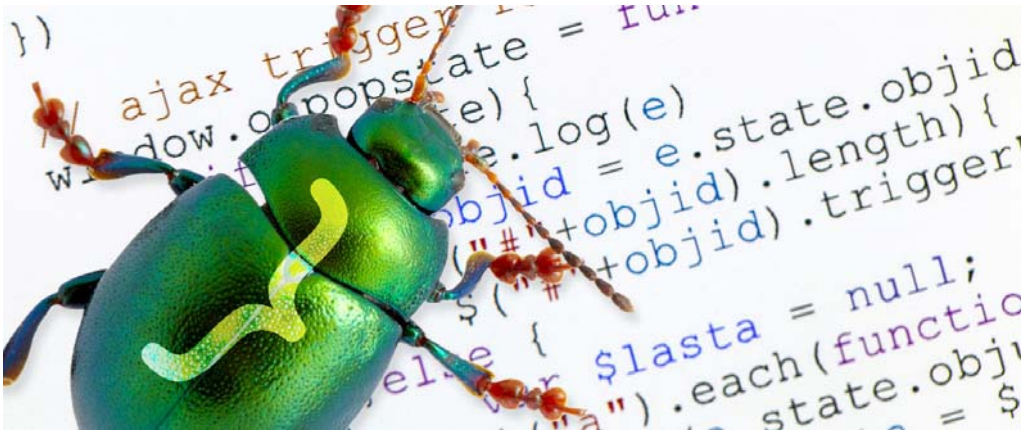


Figura 5.1: Los errores de software (bugs) se representan normalmente con la imagen de un insecto.

5.1.1.1 La depuración por puerto serie

La depuración por puerto serie, como se ha comentado en el apartado 5.1.1 [pág. 14], es una técnica de depuración ampliamente utilizada por su sencillez y conveniencia.

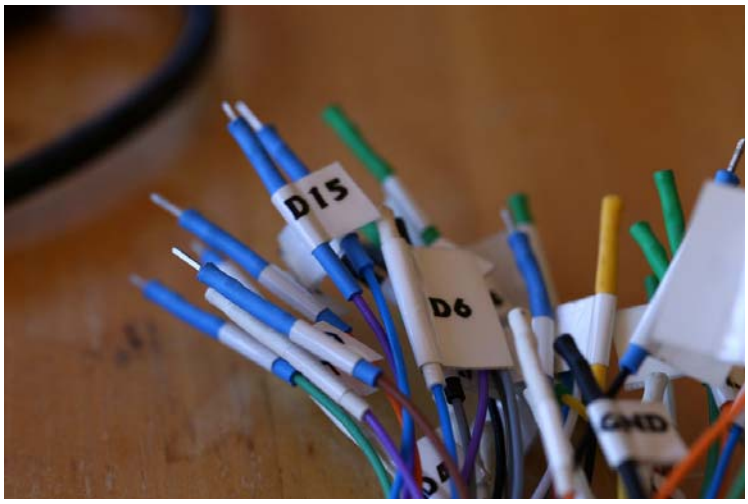
Se ejecutan una serie de comandos en el dispositivo que está siendo depurado y se mandan al puerto serie valores de variables y textos que son recogidos posteriormente por un ordenador que está escuchando ese mismo puerto serie y mostrados por pantalla. De este modo el desarrollador puede tener la certeza de que las variables de su programa toman el valor que deberían tener o que la enésima línea código ha sido ejecutada. ("Puerto Serial y Puerto Paralelo", 2016)

Estamos ante el que probablemente sea el más extendido método de depuración de programas. El estándar de puerto serie (RS-232) fue desarrollado en 1969 y posee ciertas limitaciones o características particulares (Geffrey, 2009), que se enumeran a continuación:

- Velocidades de transmisión estrictamente estandarizadas.
- Débil comprobación de errores de transmisión basadas en bits de paridad.
- Discrepancia en caracteres como el salto de línea según el sistema utilizado.
- No posee cifrado de manera nativa.
- Puede ser interceptado con suma facilidad.
- Complejidad y poca claridad para depurar más de una variable.
- Dificultad para depurar datos de múltiples fuentes en tiempo real debido a la ingente cantidad y a la poca organización de los datos que se mandan (van en serie).
- No se pueden visualizar "n" valores a la vez si no se poseen "n" puertos serie.
- Desperdicio de un 20% del ancho de banda (un paquete son 10 bits, de los que 8 son de información y 2 corresponden al bit de parada y arranque).

5.1.1.2 Depuración en el propio circuito (digital)

El otro modo de depurar un circuito digital, casi en desuso hoy en día debido a la proliferación de la tecnología digital, es la depuración "*in situ*". Consiste en realizar mediciones y monitorizaciones sobre determinadas variables físicas (medición de tensiones, por ejemplo), para posteriormente interpretarla como valor lógico. Por ejemplo, en la lógica TTL un voltaje de +5 V se corresponde a un 1 lógico y 0 V se corresponde a un 0 lógico (Lesurf, 2004).



Fuente: Flickr Static (Licencia Creative Commons).

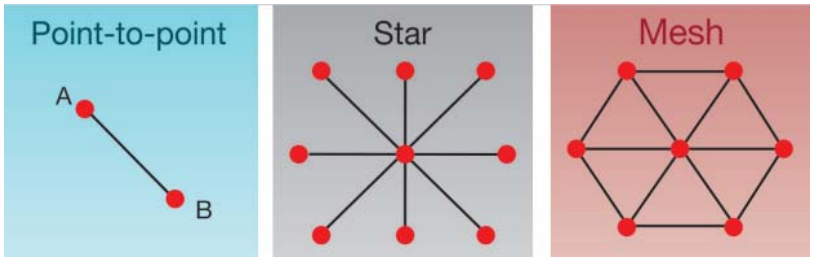
Figura 5.2: Cables etiquetados para depuración "in-circuit".

5.1.2 Interconexión de dispositivos

La interconexión de dispositivos se resuelve para la necesidad de conexión concreta del dispositivo en cuestión, pero generalmente se suelen utilizar, o soluciones no estándar de radiofrecuencia (RF) cuando es comunicación interna, o soluciones estándar (WiFi, Bluetooth™) para los casos en que la interacción con el usuario final es necesaria. De cualquier modo, la solución adoptada suele ser en la mayoría de los casos de alto consumo energético.

5.1.2.1 Topología de redes

Las redes que se suele emplear en aplicaciones de interconexión son redes P2P (de dispositivo a dispositivo) o redes con punto de acceso (infraestructura). Distinguimos varios tipos de red según su topología: punto a punto, estrella y malla (ver Figura 5.3). El modelo OSI permite configurar nuestra red con total libertad en el tipo de topología que deseemos (Bicsi, 2002; Coppley, 2015).



Fuente: (Coppley, 2015) (Licencia CC).

Figura 5.3: Tipos de redes, de izda. a dcha: punto a punto, estrella y malla.

5.1.2.2 Tecnologías de conexión

5.1.2.2.1 Tecnología Bluetooth™

La tecnología Bluetooth™ en sus distintas versiones (1.0, 2.0, 3.0, y 4.0) proporciona una conexión de corto alcance utilizando una frecuencia oscilante que varía 1600 veces por segundo (*Bluetooth Core Specification v3.0 + HS*, 2009), de modo que si hay otros dispositivos Bluetooth™, éstos no interfieren más que 1/1600 partes de la señal cada segundo, que es cuando las frecuencias se solaparían.

Es una tecnología eficaz, de corto alcance, y de alto consumo energético. Está muy extendida.

Fuente: pixnio.com (Licencia CC).
Figura 5.4: Logotipo de la tecnología Bluetooth™.

Los dispositivos se deben emparejar antes de intercambiar datos. Mediante algoritmos criptográficos se confirma que los dueños de ambos dispositivos están de acuerdo en llevar a cabo la conexión y entonces se establece la misma. Ahora ya se pueden enviar y recibir datos utilizando la gran multitud de protocolos que hay disponibles: de datos, de contactos telefónicos, de audio, de telefonía, de intercambio de archivos e incluso de conexión a internet (BTSIG, 2006).

5.1.2.2.2 Tecnologías TCP/IP y UDP

La tecnología TCP/IP es la base de lo que se conoce como "Internet". Es la tecnología subyacente de casi cualquier aplicación que trabaja a través de internet. TCP es el protocolo de envío de paquetes de uso más extendido que incluye verificación de errores de transmisión; de modo que, si un paquete se pierde o se corrompe, se puede detectar y volver a solicitar esa parte de la información. La parte "IP" se refiere a que trabaja sobre otra capa en el modelo OSI[5] (concretamente la capa de red) y es acrónimo de "Internet Protocol" (Zimmermann, 1980).

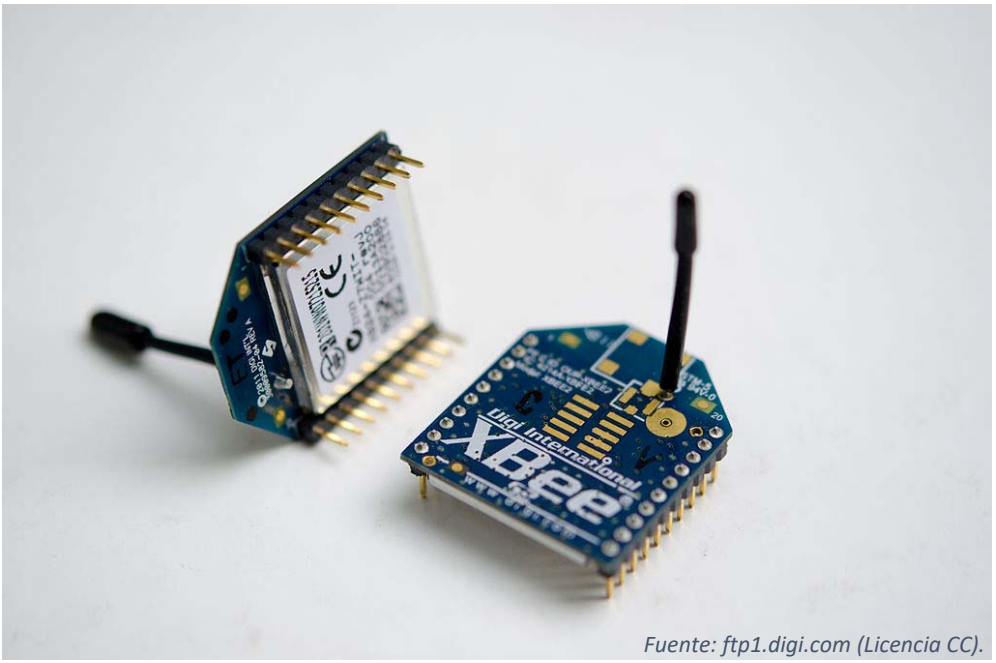
El protocolo UDP es muy similar al TCP, sólo que este último tiene la capacidad de verificar errores y el primero no (a no ser que se implemente explícitamente en nuestro programa, claro está). Como curiosidad: a pesar de que UDP también trabaja sobre IP, es muy raro (por no decir imposible) encontrar la denominación "UDP/IP". A diferencia del protocolo TCP, que sí se denomina asiduamente "TCP/IP", es una práctica extendida encontrar el protocolo UDP en bibliografía como "UDP" solamente (International Telecommunication Union, 1994).

[5] OSI es acrónimo de Open Systems Interconnection (Interconexión de sistemas abiertos).

5.1.2.2.3 Tecnología XBEE

XBEE™ es una tecnología de comunicación inalámbrica de la compañía Digi International™ compatible con la especificación "form factor" que permite las comunicaciones de corto alcance en frecuencias de 900 MHz y 2.4 GHz (Esteso, 2014). Esta tecnología consume tan poca energía como 1mW en el caso de menor consumo y 100mW en el de mayor ("Qué es un XBEE y por qué es necesario," 2014). Además de ser de bajo consumo, permite comunicaciones avanzadas multipunto (Kerry, 2014), por lo que la convierte en un candidato ideal para utilizarla en drones, donde el bajo consumo de sus componentes es un requisito bastante importante debido a que éstos trabajan fundamentalmente a batería.

Los XBEE™ aparecieron por primera vez en el año 2005 bajo el nombre de MaxStream™ alcanzando velocidades de transmisión de 250 kbits/s. (MCI Electronics, 2012)



Fuente: ftp1.digi.com (Licencia CC).

Figura 5.5: Dos módulos de radiofrecuencia (RF) XBee™.

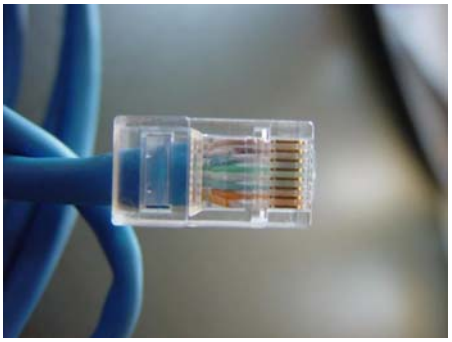
5.1.2.2.4 Tecnología Ethernet

Ethernet es un estándar de redes LAN que lleva más de 30 años desarrollándose y evolucionando. El término "ethernet" procede de "ether" (medio que los científicos del siglo XIX creían inundar todo, y por el que las ondas supuestamente se transmitían) y "net": que es el término anglosajón para "red". Ethernet define las características de cableado de redes y la implementación de los formatos de trama de datos del modelo OSI ("Historia de Ethernet", 2014).

En sus inicios, en 1972, funcionaba sobre cables coaxiales de 50 ohmios. Había problemas de reflexión de señal (limitando el máximo número de puntos de conexión de la línea) y de colisión de datos (no se comprobaba si la línea ya estaba en uso). Este ethernet se denominó "Ethernet Experimental" (o ARPANET) (Hernándo, 1991).

Más tarde en 1983 apareció el estándar IEE 802.3 o 10BASE5, que es la base del que utilizamos a día de hoy.

En 1990 se desarrolla el 10BASE-T que es el primer estándar de ethernet que no utiliza cable coaxial y utiliza cable UTP (de par no trenzado) que es muy similar al que utilizamos hoy en día en las conexiones de ethernet en ordenadores de oficinas y hogares.



Fuente: pixnio.com (Licencia CC).

Figura 5.6: Cable RJ-45 utilizado en conexiones a internet vía ethernet.

Ya en 1997 aparece la versión IEE 802.3x (una mejora del IEE 802.3) que introduce el concepto full dúplex, permitiendo recibir (RX) y transmitir (TX) datos simultáneamente, mejorando significativamente las capacidades de transmisión de datos de las redes.

Finalmente, en 2006 aparece el última y más reciente estándar: el G10BASE-T, que permite transmisiones de datos a 10Gbit/s sobre cable de par trenzado no blindado (UTP). Actualmente hay 3 especificaciones más aún en estado de "borrador".

En la siguiente figura se puede apreciar la evolución de la tecnología Ethernet en sus principales estadios de desarrollo:

Figura 5.7 Evolución de la especificación ethernet.
Fuente: Elaboración propia.

5.1.3 Comunicación y depuración de drones

La comunicación y depuración de drones difiere en algunos aspectos de la comunicación y depuración de sistemas informáticos genéricos en tanto que el requisito del medio físico de transmisión varía, y, además, aparece otra importante limitación y es que estos sistemas suelen funcionar a batería y, al estar en constante movimiento, hay que direccionar y adecuar el rango de la señal de radiofrecuencia empleada.

5.1.3.1 Sistemas de comunicación

Los sistemas de comunicación que se emplean deben ser, por lo tanto, inalámbricos. Esto implica que Ethernet ya no es una opción para comunicarnos con estos drones (salvo aquellos casos en los que nos podamos permitir anclar un cable al dron, cosa que, por norma general, no suele ser posible; quedando su uso reservado a aplicaciones muy particulares).

Estos sistemas de comunicación son soluciones particulares RF (radiofrecuencia) que no permiten por norma general funcionalidades avanzadas tales como cifrado, grabación, monitorización y visualización en tiempo real.

5.1.3.2 Sistemas de depuración

Precisamente este tipo de requisitos operacionales de los drones tratados en el apartado anterior (5.1.3.1) son los que impiden utilizar las técnicas de depuración "in-circuit" en drones. Solamente nos queda utilizar depuración remota que se hace en la inmensa mayoría de los casos por puerto serie virtual (a través de Bluetooth™ o un protocolo particular de radio frecuencia).

5.1.4 Tecnología internet of things (IoT) Sub-G

La tecnología IoT (*internet of things*), o internet de las cosas en español, es un concepto abstracto que hace referencia a la interconexión digital de objetos cotidianos con la web. Esto significa que, por ejemplo, se puede dotar a una cafetera de acceso a internet para que emita un *"tweet"* cada vez que alguien prepare café. También podemos considerar tecnología IoT unas ruedas con sensores de presión conectados a la red, que, por ejemplo, envíen un mensaje al dueño de un vehículo si la presión desciende de cierto nivel ("Cutting edge: IT's guide to edge data centers," 2016).

Según el CEO de Cisco, John Cambers, el mercado del *internet of things* alcanzará en 2030 un valor de 19 trillones de dólares norteamericanos (Kharif, 2014). De hecho, hoy en día, el 58% de los ejecutivos consultados por Cisco® Systems afirman que IoT ocupa un lugar primordial en sus estrategias de negocio más inmediatas ("Cisco Systems IoT," 2016).

El hecho de que sea "Sub-G", o "Sub-GHz" en algunas fuentes, significa que para conectarse a la red utiliza tecnología de comunicación inalámbricas de frecuencia inferior al gigahercio. Esto es de especial importancia debido a su bajo consumo, alcance de la señal aceptable y velocidad de transmisión de datos equilibrada. Además, las bandas de frecuencia inferiores al gigahercio no están tan sobresaturadas de señales como las de la banda 2.4 GHz, utilizadas por dispositivos WiFi y Bluetooth™ primordialmente, aunque no exclusivamente (Coppley, 2015).

5.2 PLANTEAMIENTO

5.2.1 Definición exhaustiva y extensiva de los requisitos del problema

El problema que aquí se plantea, tal y como ya se ha explicado sin entrar en detalle en la introducción del presente informe, es desarrollar una plataforma digital que cumpla con una serie de características básicas y que permita interconectar dispositivos electrónicos inteligentes (IoT).

La solución final ha de ser capaz de obtener datos de dispositivos inteligentes en tiempo real y éstos han de ser accesibles desde un simple navegador de internet (previa autenticación con credenciales de usuario). Asimismo, también debe emplear distintas tecnologías de conexión para que sea adaptable a todas las posibles necesidades particulares de proyectos basados en IoT, dispositivos electrónicos, o drones; luego ha de ser accesible desde teléfonos móviles inteligentes (*smartphones*) y tabletas digitales. Finalmente debe ser capaz de proporcionar las medidas de seguridad que permitan proteger a la red de ataques de piratas informáticos. La siguiente figura (Figura 5.8) puede ayudar a aclarar el concepto:

Fuente: Elaboración propia.

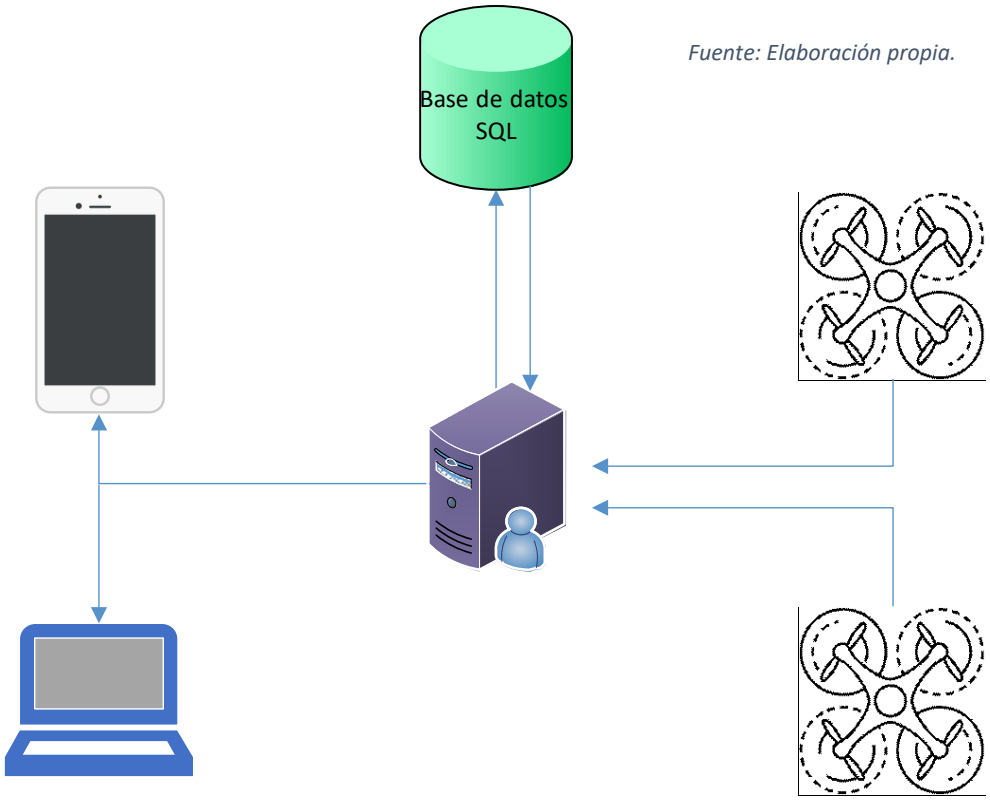


Figura 5.8: Diagrama simplificado de interconexión de dispositivos.

Ésta plataforma ha de ser capaz, en última instancia, de solucionar los problemas de nula estandarización que caracteriza a la mayoría de dispositivos IoT, a la par que ha de facilitar significativamente las tareas de desarrollo y programación (en su fase de depuración de errores e implementación).

La plataforma deberá también ser compatible con los principales lenguajes de programación, navegadores, teléfonos móviles y dispositivos IoT. Existe pues, un gran trabajo de generalización de código y de programar diversas veces las mismas funciones, pero adaptada a todas y cada una de las distintas plataformas y tecnologías. En concreto:

Plataformas IoT soportadas:

1. Arduino™
2. Genuino™
3. Arduino-compatible
4. ARM® Microcontrollers
5. C-Compliant Target Platform

Plataformas informáticas soportadas:

1. Sistemas basados en UNIX® con compilador GNU C/C++ 4.9.2+
2. Microsoft® Windows™ Vista o superior
3. Microsoft® Windows™ 98, 2000/ME, XP (no recomendado)

Navegadores soportados:

1. Google Chrome™ (Recomendado)
2. Mozilla Firefox™
3. Microsoft Internet Explorer® 10+ / EDGE®

Smartphones soportados:

1. iPhone® 4+ (Safari™)
2. Android® v3.5+ (Google Chrome™)

Además, se quiere que este sistema esté aplicado al control de drones. Es decir, a pesar de ser generalizable a cualquier dispositivo electrónico que sea capaz de procesar y mandar señales, y ejecutar código C, se le va a dar una plataforma un enfoque exclusivo al control de drones. Esto implica que se la va a dotar de funcionalidades específicas para el control y monitorización de drones.

Esto último no implica que no se pueda utilizar para cualquier dispositivo electrónico, aunque éste no sea un dron. Debido al potencial de una plataforma de estas características, se ha optado por, en primer lugar, desarrollar la plataforma orientada a drones, que es la que nos ocupará en este proyecto, y, en segundo lugar, de cara a un futuro, generalizarla ligeramente (aunque las modificaciones serían mínimas) para dar cabida a todo tipo de dispositivos.

Cabe decir que en la plataforma orientada a drones ya se pueden utilizar todo tipo de dispositivos, aunque estos no sean drones; lo que ocurre es que hay algunas herramientas que están enfocadas a drones y tanto los textos como el nombre de la plataforma (DroneLink™) ya hablan de “drones”.

En principio la plataforma está diseñada para que la comunicación sea bidireccional. Es decir, de la plataforma al dron y del dron a la plataforma. En una primera fase, se ha implementado la comunicación en una sola dirección para, posteriormente, refinar el producto final y con pocas (y muy ligeras modificaciones, que implican más un trabajo de interfaz de usuario que de lógica) dar el salto a la bidireccionalidad.

La empresa FutureSiSens®, ganadora de varios reconocimientos y patrocinada por Repsol®, se ha interesado recientemente por la presente plataforma de comunicación en tiempo real para adaptarla a sus desarrollos particulares.

5.2.2 Relación con las tecnologías existentes

En este apartado se presenta la relación que guardan las tecnologías expuestas en el apartado de estado del arte (5.1) con el proyecto que nos ocupa.

En primer lugar, cabe destacar que ciertas de las tecnologías expuestas en el apartado 3.1, se utilizan de manera implícita. Esto significa que cuando se utiliza una librería para mandar un paquete TCP sobre ethernet, ya estamos utilizando el protocolo TCP/IP sin necesidad de conocer exactamente los detalles más recónditos del mismo.

El Bluetooth™ y el XBee™ son protocolos de comunicación inalámbrica que pueden utilizar los dispositivos para acceder a la API.

5.2.3 Partes diferenciadas de la plataforma

La plataforma es un sistema complejo que combina muchísimas tecnologías que trabajan juntas persiguiendo un único fin: la comunicación en tiempo real. Por ello es conveniente definir bien las distintas partes de ésta y la relación que guardan entre ellas.

El siguiente esquema (Figura 5.9) permite al lector formarse una mejor imagen del funcionamiento externo de la plataforma, esto es, el flujo de los datos y su dirección. Este esquema hace referencia a dos drones conectados mediante tecnologías distintas a la plataforma, y programados en dos lenguajes distintos, actualizando variables en la plataforma. La información es simultáneamente consultada por un portátil y un teléfono inteligente (o *smartphone*, en inglés).

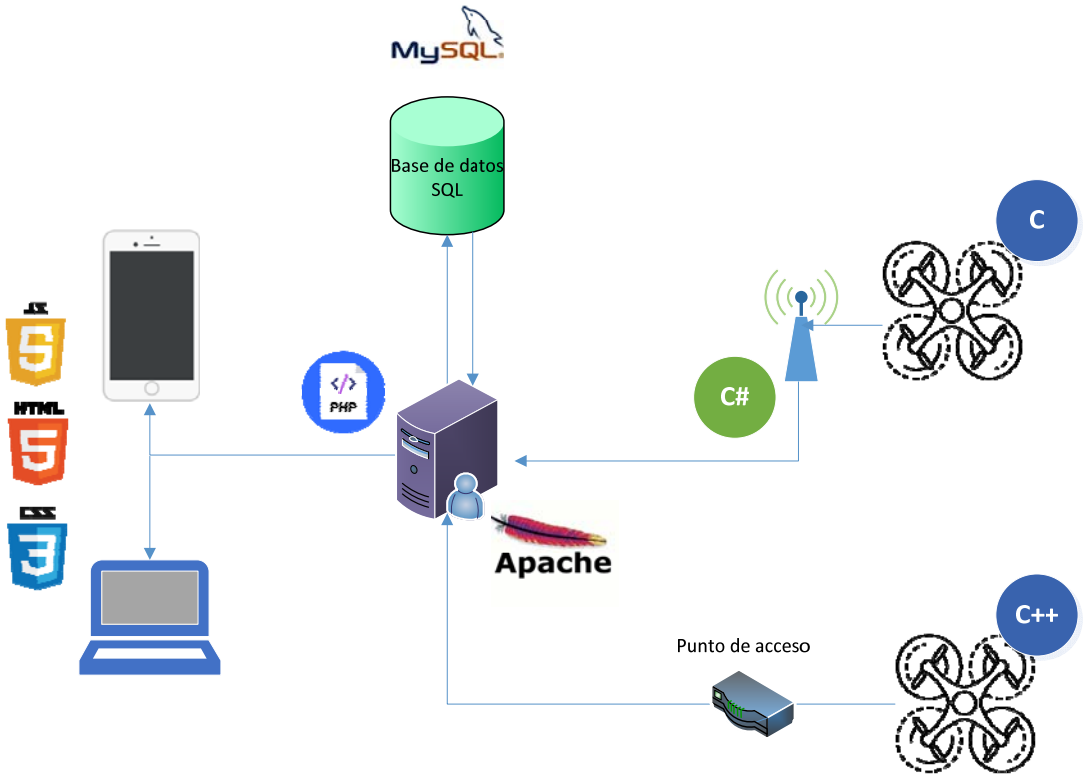


Figura 5.9: Diagrama simplificado de interconexión de dispositivos y tecnologías subyacentes de cada fase. Fuente: Elaboración propia.

Tal y como puede apreciarse en la figura anterior (Figura 5.9), la plataforma DroneLink™ consta de cuatro partes diferenciadas que cooperan entre sí para alcanzar su propósito final.

En la siguiente figura (Figura 5.10) se pueden apreciar las distintas partes fundamentales de la plataforma:

Fuente: Elaboración propia.

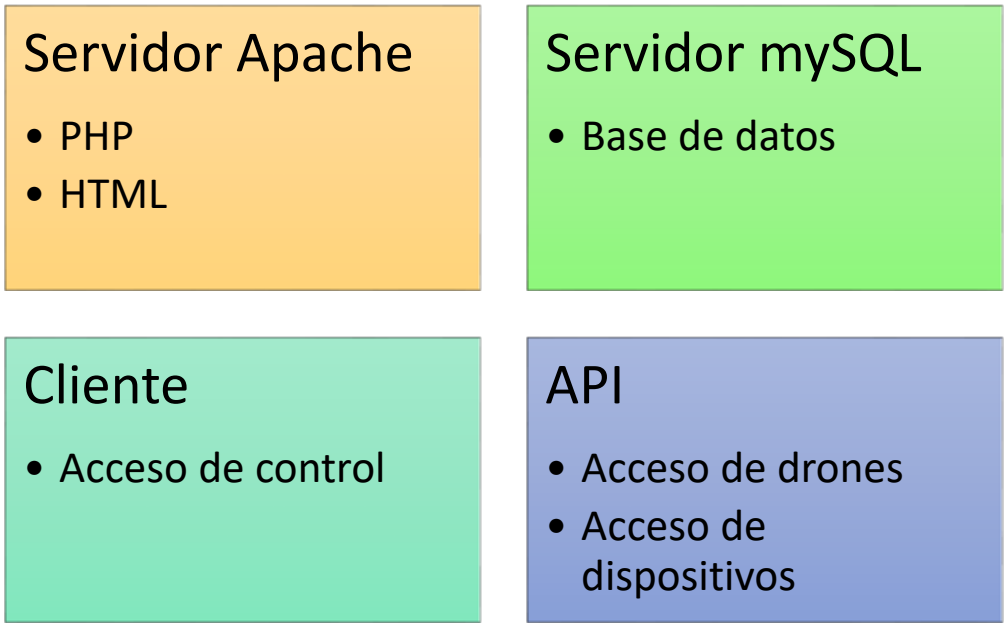


Figura 5.10: Partes diferencias de la plataforma.

Entrando un poco más en detalle en las distintas partes que conforman la plataforma, se exponen a continuación las implementaciones de cada parte:

1. El servidor web (implementado en Apache™ pero pudiéndose utilizar otro)
2. La base de datos SQL, implementada en mySQL.
3. La API (que permite a los drones o dispositivos acceder al servidor), en PHP.
4. La lógica de cliente (ordenadores y *smartphones*) que acceden a la aplicación, en HTML5, JavaScript, JQuery, y CSS3.

La primera parte se encarga de procesar las peticiones de drones y clientes, actualizando la base de datos y consultando información de ella para posteriormente mandarla a quien sea pertinente. Actúa como núcleo de procesamiento, esto es, donde reside el corazón de la plataforma.

Dentro de la primera parte diferenciamos, a su vez, el servidor *http* de páginas HTML5, y el servidor PHP. El primero se encarga de gestionar las peticiones entrantes (HTTP, XHR) y del "estilo" de las páginas (texto fijo, formas, colores, diseños, imágenes, etc.) y el segundo de la lógica del servidor (variables, gestión de drones, usuarios, datos, etc.).

La segunda parte se encarga de almacenar, procesar y recuperar los datos del disco duro del servidor. Sólo se comunica directamente con el servidor PHP, encargado de gestionar los datos.

La API (*Application Programming Interface*) se encarga de proporcionar un punto de acceso programable para los dispositivos inteligentes y/o drones que deseen conectarse a la plataforma. Este punto engloba a toda la comunicación entre el dron y el servidor (estadios intermedios incluidos).

Finalmente, la lógica de cliente es mandada al cliente por el servidor "*http*" y se encarga de actualizar datos en tiempo real mediante peticiones *AJAX* sobre *XHR*, presentar los datos elegantemente, y recoger las entradas del usuario. Conforma la denominada GUI (*Graphical User Interface*) o interfaz gráfica de usuario. Sírvase el lector de consultar la siguiente figura (Figura 5.11), donde se muestra la pirámide tecnológica de la plataforma:

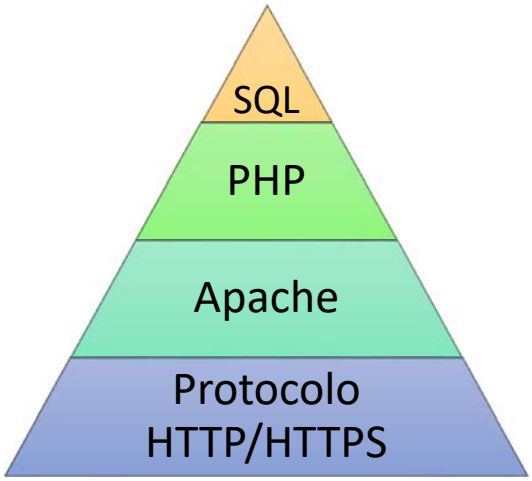


Figura 5.11: Pirámide de dependencias tecnológicas.
Fuente: Elaboración propia.

Podemos observar como el servidor mySQL interacciona con PHP, que interacciona a su vez con Apache, que finalmente manda sobre protocolo HTTP o HTTPS el resultado final de la petición.

5.2.4 Procedimientos estándar de manipulación de datos

Algunos procedimientos o flujos de trabajo principales son fundamentales por estar situados en el núcleo de la plataforma, por su utilidad, y por la cantidad de veces que se utilizan (sensiblemente mayor a ningún otro proceso de este software).

Existen dos de éstos procesos; a saber: consulta de una variable y escritura de una variable.

5.2.4.1 Consulta de una variable del sistema DroneLink™

Al consultar una variable (inicialmente por parte del cliente, en un futuro los dispositivos podrán consultarlas también utilizando este mismo flujograma) se debe seguir el proceso indicado en la siguiente figura (que cumple el estándar UML):

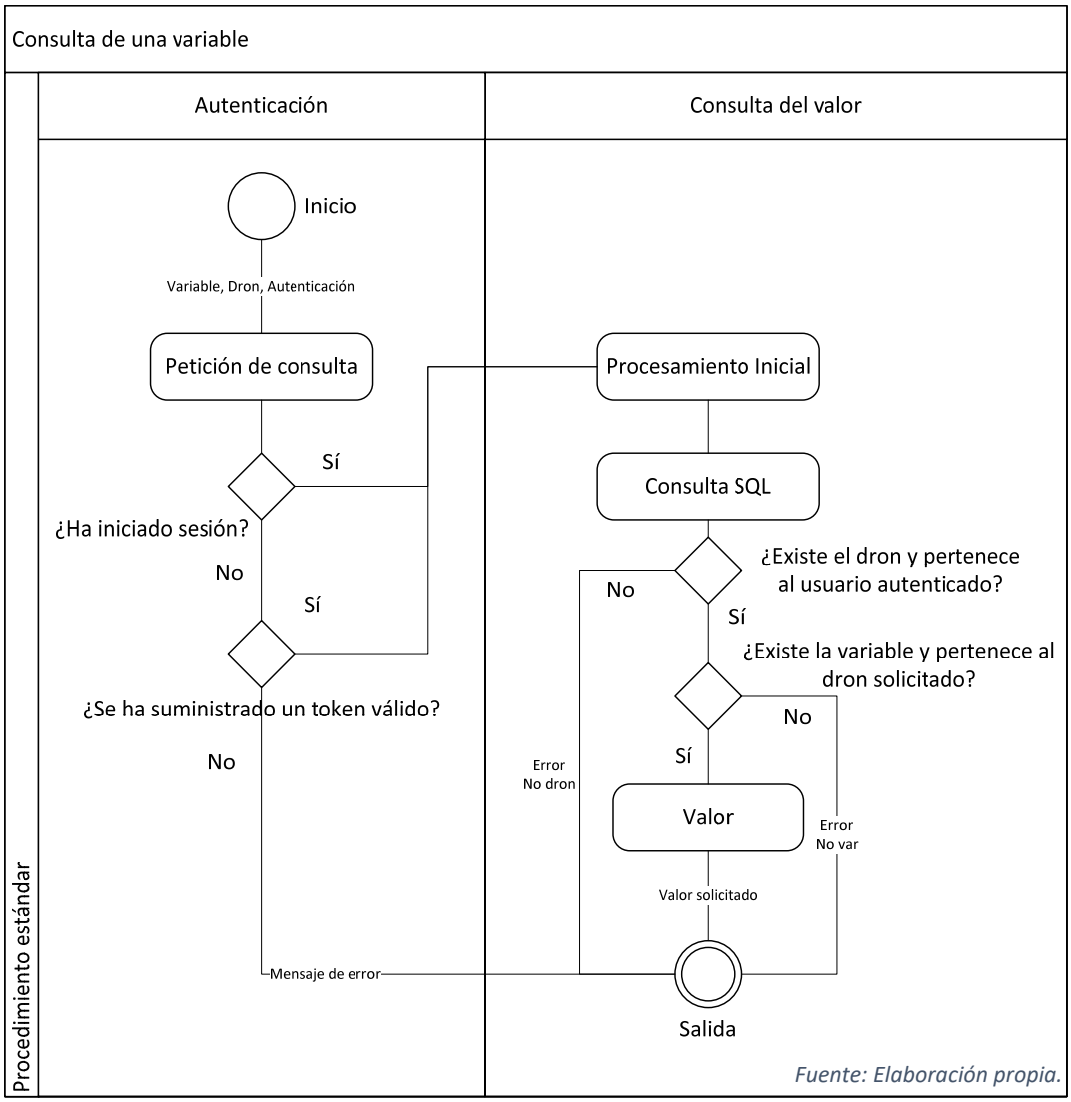


Figura 5.12: Diagrama UML de actividades, caso de consulta de una variable.

Lo que apreciamos en la figura 11 es el procedimiento seguido por el programa desde que se inicia una solicitud de obtención del valor de una variable hasta que ésta se suministra (o no). Se hacen comprobaciones seguras de autenticación (nombre de usuario y contraseña cifrada con SHA1), y de coherencia interna, es decir, que el dron pertenezca al usuario que solicita y que la variable pertenezca al dron solicitado.

5.2.4.2 Actualización o modificación del valor de una variable del sistema DroneLink™

El siguiente diagrama muestra el flujo de trabajo de la plataforma al modificar el valor de una variable. Esta acción suele (en la fase inicial de la aplicación) solicitarla el dron al servidor exclusivamente. El siguiente diagrama estándar UML ilustra este procedimiento:

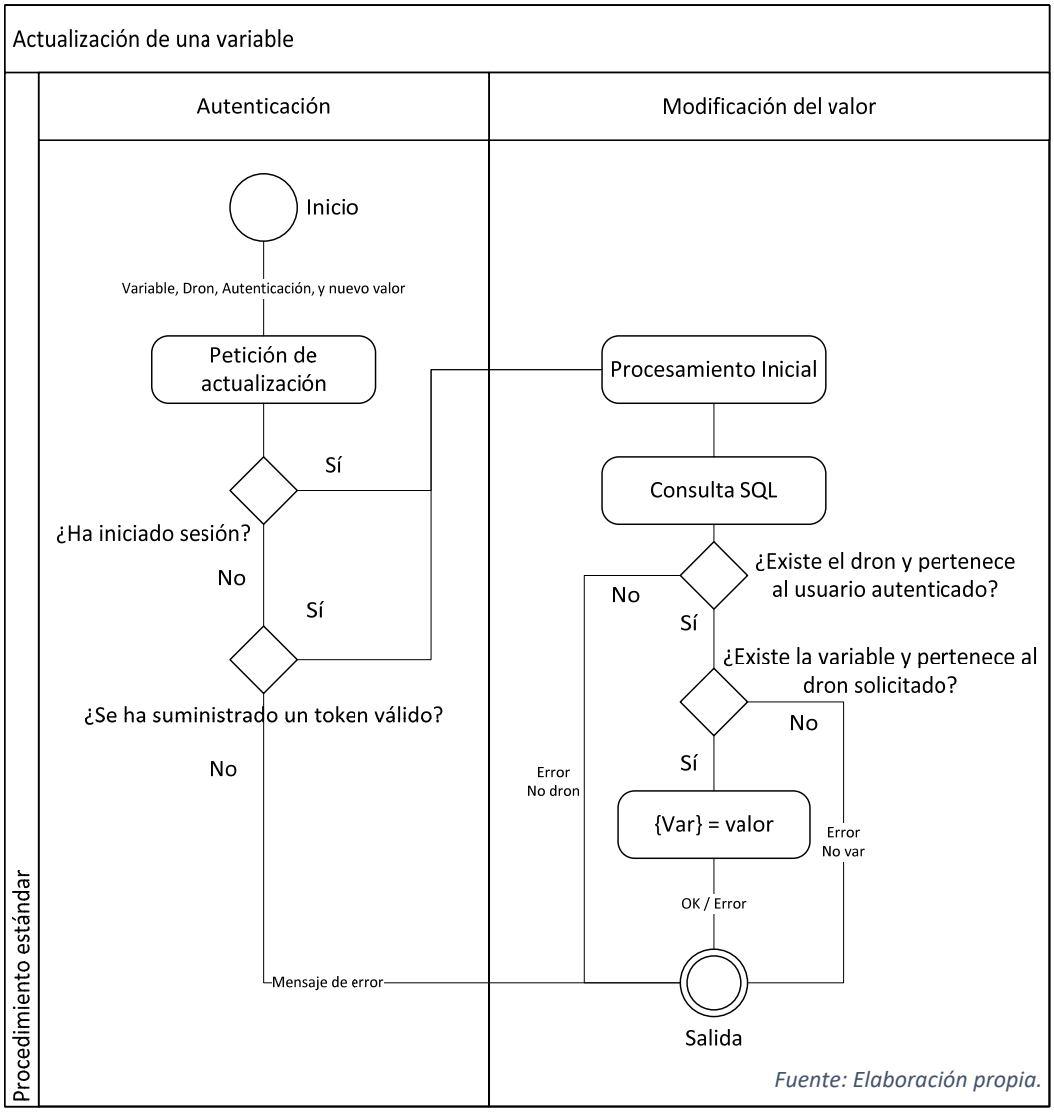
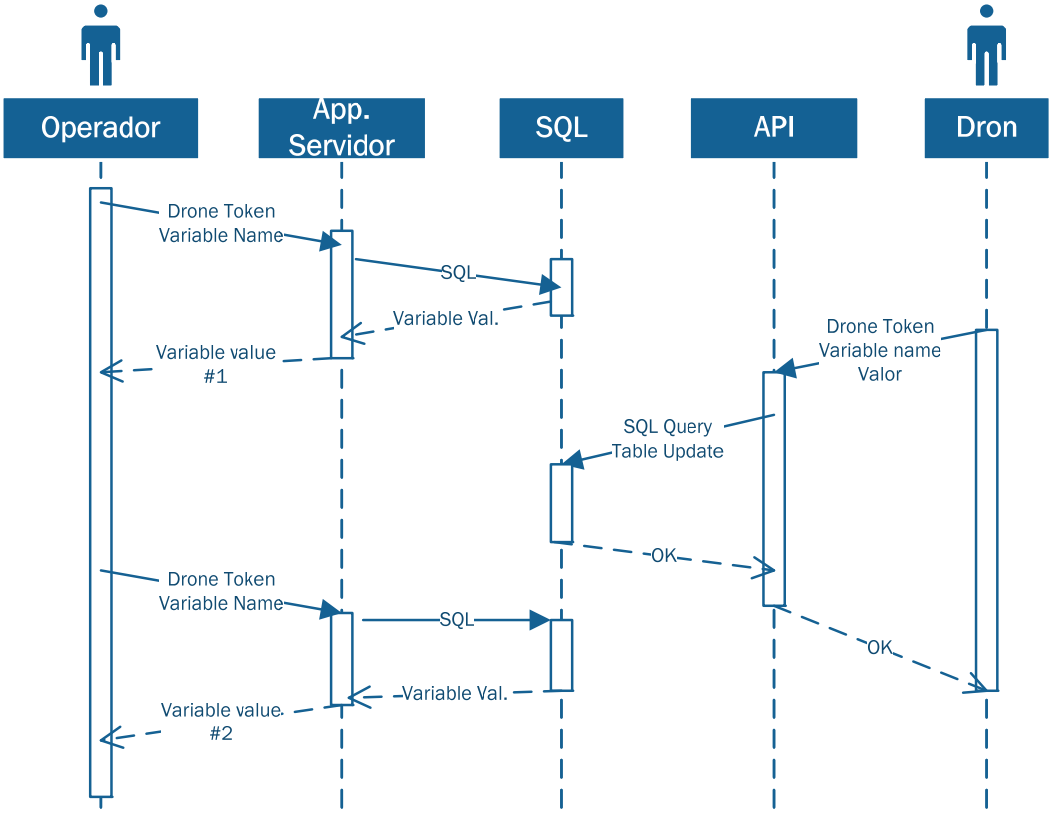


Figura 5.13: Diagrama UML de actividades, caso de actualización de una variable.

Se puede observar que en este caso necesitamos suministrarle al sistema un dato más: el nuevo valor que queremos escribir en la variable en cuestión.

5.2.4.3 Diagrama de secuencias de una lectura y actualización en la plataforma

El siguiente diagrama UML muestra el funcionamiento interno de la plataforma cuando un cliente accede a los datos del servidor, a continuación, un dron actualiza la información del servidor, y finalmente el cliente (u operador) vuelve a leer el valor de dicha variable.



Fuente: Elaboración propia.

Figura 5.14: Diagrama UML de secuencias de una lectura y actualización en la plataforma.

En este diagrama se supone, naturalmente, que el cliente ya está autenticado en el sistema (se le ha concedido acceso previamente). El dron, por el contrario, debe autenticarse cada vez que efectúa una petición.

5.2.5 Justificación de la elección de la tecnología de implementación

Vistos los requisitos de interoperabilidad y compatibilidad del sistema se necesitaba asentarlo sobre una serie de plataformas lo suficientemente flexibles, compatibles entre sí, y compatibles con el mayor número de dispositivos.

Al basar el software en un servidor no solamente se gana accesibilidad, sino que también se gana una cierta tranquilidad puesto que la parte importante del programa funciona sobre un entorno que podemos controlar nosotros (ahí donde se ejecuta el programa principal) y que sólo tiene entradas y salidas. A cambio, sacrificamos algo de velocidad puesto que un servidor HTTP que atiende peticiones puede según cómo tardar en responder (en función de la distancia física, de la ruta de conexión de internet, etc.).

Fuente: Elaboración propia.

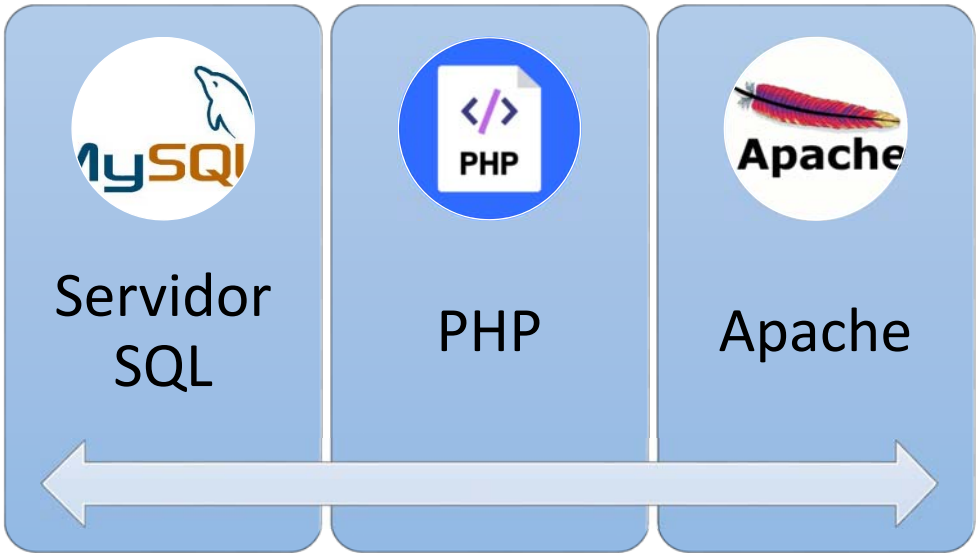


Figura 5.15: Diagrama de flujo de datos (lógica interna del servidor).

El sistema Apache™, sin embargo, no permite ejecutar consultas SQL en una base de datos (o como mínimo no mediante procedimientos habituales), y tampoco puede procesar scripts o lógica de lado de servidor. Lo único que puede hacer es mandar documentos de hipertexto (HTML) con algo de lógica de lado de cliente (en lenguaje JavaScript, abreviado JS), la cual es muy útil, pero no podemos depender de ella puesto que se ejecuta en el lado del cliente con las vulnerabilidades de seguridad que ello implica: el cliente puede ver todo el código y manipular su ejecución si es un usuario avanzado. Reservaremos el código JS para comprobaciones redundantes, peticiones de JSON empleando AJAX sobre XHR (ya veremos lo que es esto más adelante, suena complicado, pero en realidad es muy simple) y cuestiones de estilo y formato.

Este problema lo atajamos adjuntando otra capa de procesamiento en el servidor Apache™ llamada PHP. Las siglas PHP (*Hypertext pre-processor*) significan preprocesador de hipertexto y es esta tecnología la que se encarga de interactuar con la base de datos SQL y ejecutar scripts interpretados

de lógica de servidor. El resultado se presenta en forma de HTML y se manda a Apache™ para que éste lo envíe al cliente (ver Figura 5.15).

Para la base de datos se ha elegido SQL (*Structured Query Language*) por ser el estándar más utilizado mundialmente en consultas de datos y mySQL™ (variante de SQL) en particular porque tiene una sintaxis más clara, aunque menos poderosa que otras cómo Oracle® SQL, por ejemplo. Sirve de "buffer" intermedio entre el cliente y el servidor a la par que almacena todos los datos, preferencias, drones, variables y contraseñas de los usuarios de la plataforma.

Para el lado del cliente se ha optado por utilizar HTML5 + CSS (lenguajes web) por ser la especificación más actual, dominante y compatible que existe hoy en día. Se ha utilizado una librería llamada Bootstrap (desarrollada originalmente por el equipo de Twitter® y liberalizada posteriormente) para mejorar notablemente la distribución de la información en pantalla. De diseño base, se ha utilizado la plantilla AdminLTE™, que a su vez utiliza todas las tecnologías acabadas de mencionar, puesto que es gratuita y permite desarrollar aplicaciones de servidor con suma facilidad.

En la siguiente figura se puede apreciar los logotipos de estas tecnologías:

Fuente: Trabajo derivado a partir de obras originales de W3C. Imágenes originales bajo licencia CC.

Figura 5.16: Logotipos de las principales tecnologías de desarrollo web. De mayor a menor importancia y de izquierda a derecha y de arriba a abajo: HTML5, JS, CSS3, Bootstrap.

5.2.6 Herramientas utilizadas para la implementación de la plataforma

Para implementar la plataforma se han utilizado diversas herramientas que se adecuan a la parte de la misma que se desarrolla. A continuación, se enumeran las partes de la plataforma y se explica qué herramientas se han utilizado en cada una.

5.2.6.1 Lado de cliente de la plataforma

Para implementar el lado de cliente de la plataforma se ha optado por Aptana® Studio 3 (software libre de edición web) y una buena alternativa al tradicional NVU KompoZer 0.8b3, proyecto abandonado hace ya 5 años, y que se consideró utilizar inicialmente. Este programa (Aptana® Studio 3) permite la edición de código HTML5 (visualización) y JavaScript (lógica).

Fuente: Elaboración propia.

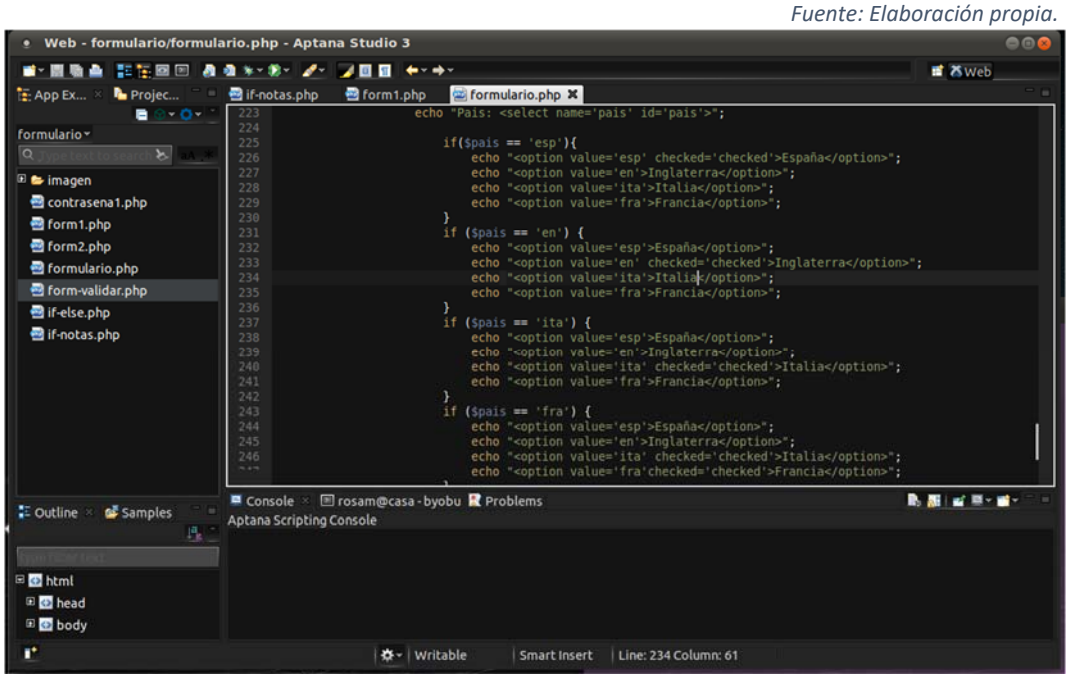


Figura 5.17: Captura de pantalla del programa Aptana® Studio 3.

Se ha utilizado un servidor de pruebas que incluye Apache™, PHP y mySQL en una sola instalación. El paquete en cuestión se llama XAMPP™ y sirve para probar webs antes de que sean instalados en un entorno de producción (servidor profesional).

Fuente: XAMPP Project (Licencia CC)
Figura 5.18: Logotipo del paquete de programas XAMPP™.

Finalmente se ha utilizado Google® Chrome™ para probar y depurar la plataforma. En concreto las herramientas de depuración que provee Google® Chrome™ para este propósito llamadas "Developer Tools".

5.2.6.1.1 Herramientas de depuración

Para programar las herramientas de depuración se ha utilizado Processing™ 3, que es un software de representación gráfica basado en Java™ de Oracle®. La herramienta que se ha desarrollado es el serial software plotter. Esta herramienta permite monitorizar el puerto serie en el formato de comunicaciones (protocolo) utilizado por DroneLink Bridge™ (ver apdo. 5.2.6.3.1) y graficar en pantalla y guardar copia de toda la información en el disco duro en formato Microsoft® Excel™.

La siguiente figura (Figura 5.19) muestra el software serial plotter en ejecución y a punto de iniciar una nueva página en blanco y guardar la página actual (ya casi llena) en un archivo Microsoft® Excel™:

Fuente: Elaboración propia.

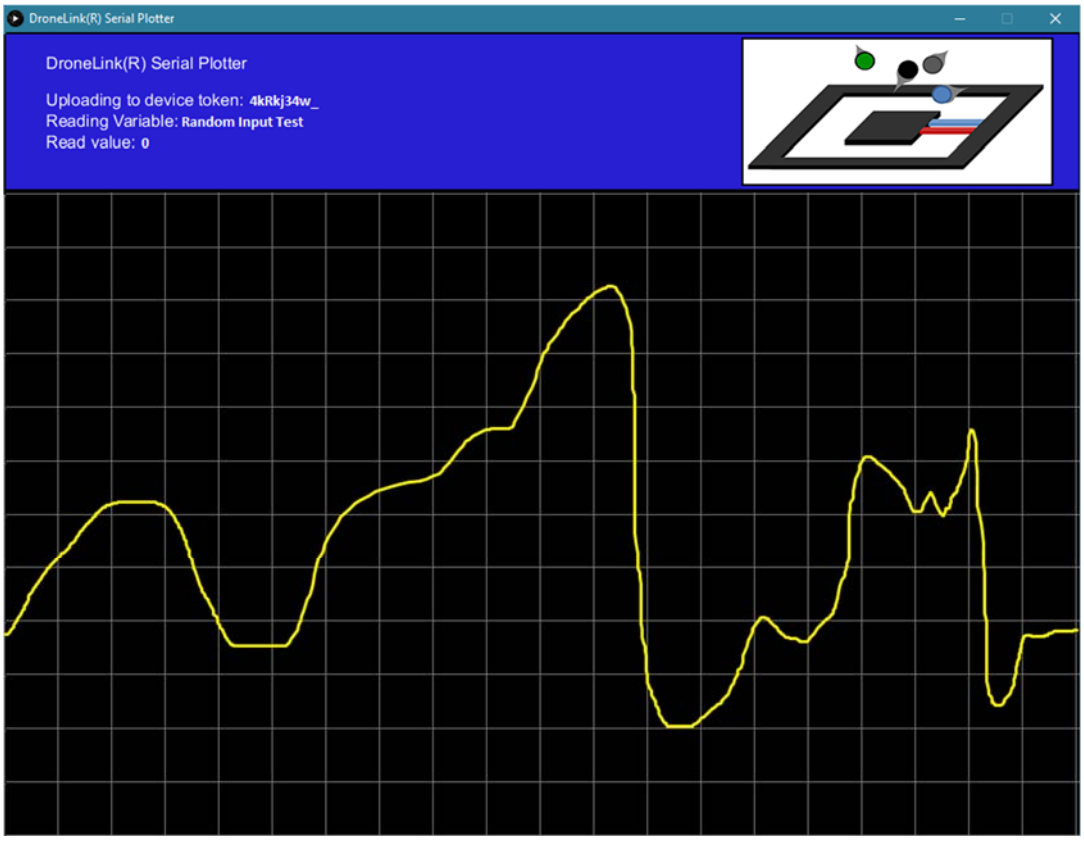


Figura 5.19: Software serial plotter capturando una señal analógica en formato DroneLink®

5.2.6.2 Lado del servidor

Del lado del servidor se encarga el código PHP, y para desarrollarlo se ha utilizado Aptana® Studio 3. El código del lado de servidor se ha depurado mediante el método ensayo-error (utilizando mucho la tecla F5) y utilizando la función *var_dump()* nativa de PHP.

5.2.6.3 Programación de la API

5.2.6.3.1 Programación de la solución DroneLink Bridge™

DroneLink Bridge™ es un subprograma implementado perteneciente al conjunto de la plataforma que se encarga de ejercer de pasarela o de punto de acceso intermedio entre el dispositivo o dron y la plataforma DroneLink™. Está programado íntegramente en C# y trabaja sobre puerto serie virtual (Bluetooth™ o XBee™) y Ethernet o WiFi. Básicamente se encarga de captar las señales Bluetooth™ o XBee™ enviadas por el dispositivo, procesarlas, y subirlas a un servidor con el software DroneLink™ ejecutándose en él.

Fuente: Elaboración propia.

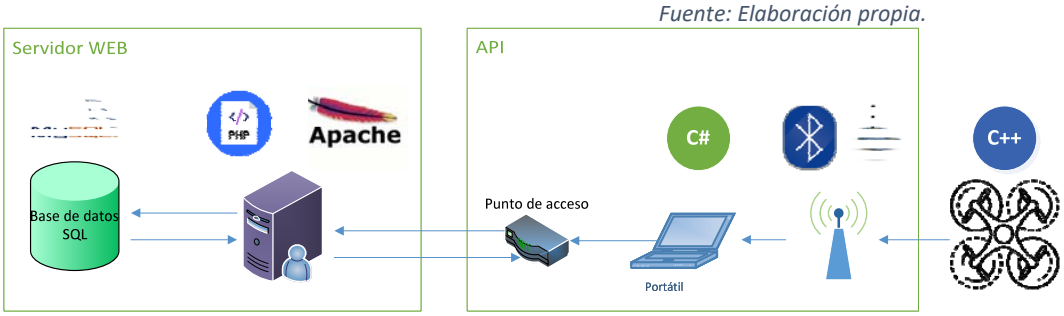


Figura 5.20: Diagrama detallado de una conexión a la plataforma a través de la API.

5.2.6.3.2 Programación de las librerías ethernet

Estas librerías proporcionan conexión directa de cualquier dispositivo o dron (aunque se desaconseja el uso para estos últimos por razones evidentes) directamente a internet, sin necesidad de utilizar un ordenador como intermediario. Necesitan cableado físico del dispositivo al enrutador.

5.2.6.4 Programación y desarrollo de la base de datos SQL

Para esta tarea se ha utilizado una herramienta muy versátil que corre sobre código PHP que se llama "phpMyAdmin". Proporciona capacidades de edición, creación de tablas, ejecución de consultas, gestión de usuarios y depuración de errores.

Fuente: phpMyAdmin Project. Licencia CC.
Figura 5.21: Logotipo del programa phpMyAdmin™.

5.2.7 Aspectos de seguridad e interfaz de usuario (GUI)

La seguridad es un aspecto que se ha cuidado en todo momento, solicitando internamente credenciales y datos de sesión constantemente durante la ejecución de la aplicación y comprobando y saneando todas y cada una de las entradas manuales del usuario final para evitar ataques informáticos como inyecciones SQL y o XSS.

Se debe introducir también un sistema de registro de los intentos de sesión fallidos, de modo que una cuenta de usuario quede bloqueada si se producen más de 3 inicios de sesión fallidos. El sistema deber registrar, entre otras cosas, la dirección IP de la que proviene el intento de inicio de sesión fallido.

Para la interfaz de usuario es conveniente utilizar una plantilla HTML. Trabajando a partir de la plantilla se evitan muchas complicaciones y problemas que van más allá del objetivo del presente trabajo y que podrían llegar a evitar la consecución de los mismos.

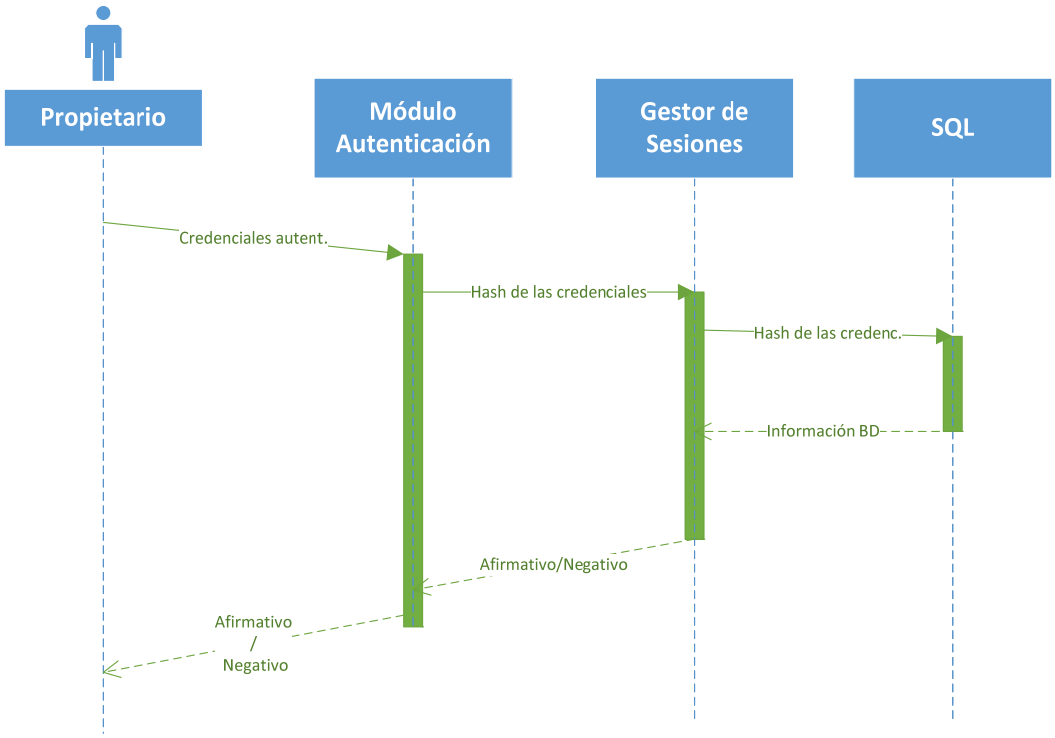
La seguridad, en definitiva, no es un complemento, sino un objetivo principal, y una de las características que diferencia a esta solución de las técnicas empleadas hasta la fecha.

5.2.7.1 El inicio de sesión

El inicio de sesión es un aspecto que se ha cuidado mucho y consta de unas funciones programadas específicamente para la ocasión que mejoran notablemente la funcionalidad por defecto de PHP, evitan ataques SQL y XSS directos y reflejados. El robo de cookies tampoco es posible.

Adicionalmente, las contraseñas se almacenan en la base de datos mediante técnicas de cifrado criptográfico (con *hash* y *salt and pepper*) que hacen que en el eventual (y difícil) caso de que un pirata informático consiguiera acceder a la base de datos SQL, en el apartado de contraseñas no vería nada más que números y letras sin sentido aparente (son las contraseñas cifradas).

La autenticación puede describirse mediante el siguiente diagrama de secuencias UML:



Fuente: Elaboración propia.
Figura 5.22: Diagrama UML de secuencias, caso de una autenticación por parte de un usuario.

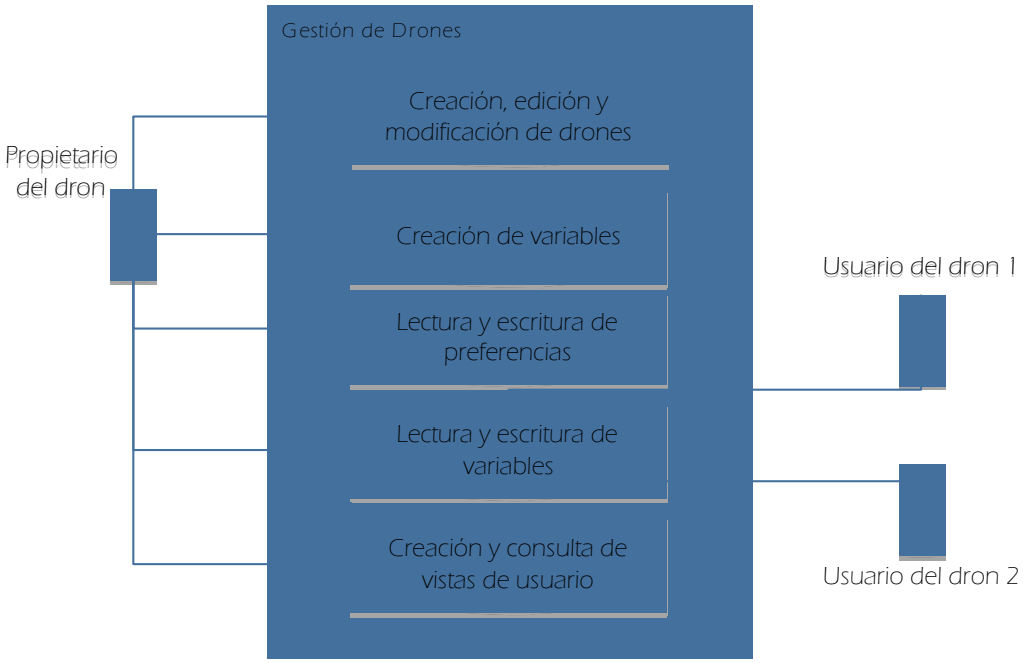
5.3 DIAGRAMAS ESTÁNDAR UML DE LA SOLUCIÓN

Se han desarrollado diversos diagramas UML para describir comportamientos específicos de partes del software, debido a la imposibilidad de incluir en diagramas todo el software por problemas de extensión, se incluyen las partes más significativas y generales en este apartado.

Hay diagramas UML que no están expuestos en este apartado porque hacen referencia a partes concretas de la aplicación y se exponen en el apartado pertinente.

5.3.1 Diagrama de casos de uso

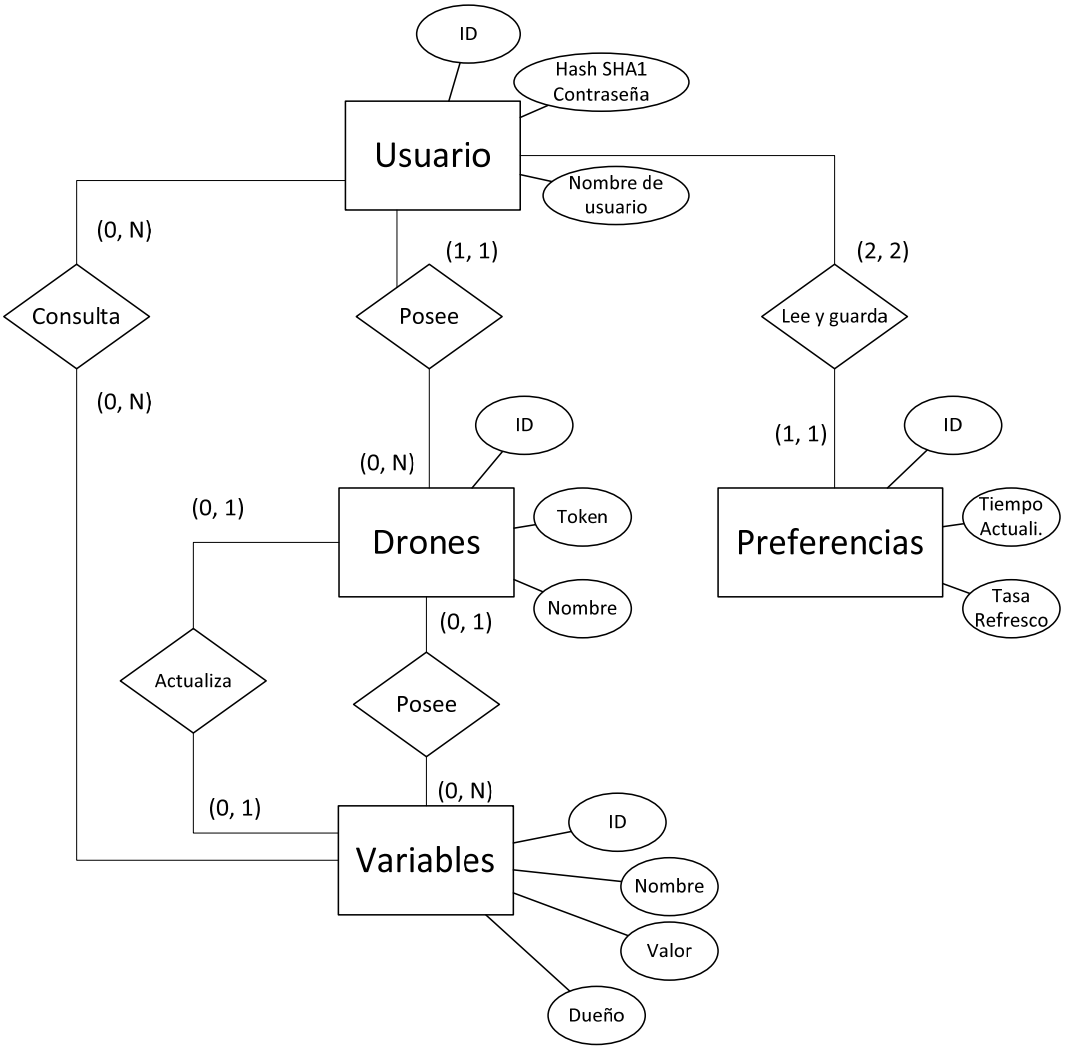
En este diagrama podemos apreciar los distintos casos de uso de la plataforma; siendo éste uno de los diagramas UML (*Unified Modeling Language*) o lenguaje de modelado unificado según su acrónimo en inglés.



Fuente: Elaboración propia.
Figura 5.23: Diagrama UML de casos de uso de la plataforma.

5.3.2 Diagrama de entidad/relación o de Chen simplificado con cardinalidades

En el siguiente diagrama estándar UML de E/R se puede ver la relación y las cardinalidades de las entidades principales que conforman la plataforma. El sistema completo es muchísimo más complejo; esta vista es una simplificación con las entidades y atributos más relevantes.



Fuente: Elaboración propia.
Figura 5.24: Diagrama UML de entidad/relación simplificado con las cardinalidades indicadas.

5.4 DESARROLLO DE LAS SOLUCIONES

5.4.1 La base de datos

La base de datos es una parte crucial de la plataforma, pues es la pieza central que interconecta las principales partes del rompecabezas. Como ya se ha explicado, se ha utilizado el sistema mySQL por su simplicidad, gratuidad, y velocidad de trabajo.

Los datos se almacenan en el disco duro del servidor, lo cual es una operación E/S bloqueante y costosa y quizás no es el mejor método para aplicaciones en tiempo real que necesiten funcionar a más de 10Hz (ciclos de lectura/escritura), pero suficiente para la aplicación y el propósito propuesto en los objetivos del proyecto.

5.4.1.1 Requisitos de gestión de información

Para diseñar la arquitectura de una base de datos, lo primero que hay que tener en cuenta es la información que se va a almacenar para decidir cómo vamos a organizarla. Aunque naturalmente este es el punto más importante, hay otros que no deben descuidarse como: quién va a acceder a esa información, con qué frecuencia y desde dónde va a acceder. Aquí se resuelven estas cuestiones una a una.

La base de datos ha de ser capaz de gestionar la siguiente información:

- Usuarios registrados en el sistema y credenciales
- Preferencias de usuario
- Mensajes de depuración
- Drones registrados y a qué usuarios pertenecen
- Variables registradas y a qué dron pertenecen
- Intentos de inicio de sesión fallidos
- Vistas de usuario
- Grabaciones de datos

Para gestionar dicha información, se ha decidido crear una tabla por categoría, obteniendo 9 tablas en total.

Luego está la cuestión de quién va a acceder a la información. La respuesta es tan simple como concisa: la plataforma y la *API*. En cualquier caso, siempre peticiones que provienen de dentro del propio servidor (en concreto del módulo PHP) por cuestiones de seguridad. De por seguro sería más rápido conectarse al servidor SQL y lanzar la consulta externamente, pero esto conllevaría problemas de seguridad terribles, puesto que en primer lugar la contraseña interna del servidor SQL debería almacenarse externamente y, en segundo lugar, ¿quién garantiza que un usuario malintencionado no mandaría consultas maliciosas al servidor SQL?

La información debería poder obtenerse como mínimo una vez por segundo (≥1 Hz) y debería poder ser accesible desde todo el mundo, pero pasando por el servidor Apache-PHP primero.

5.4.1.2 Tablas y motor de base de datos SQL

Las tablas que se han creado se corresponden con la lista de requisitos de gestión de la información expuesta en el apartado anterior, con un par de salvedades. Las tablas y los campos que se han creado son los que se observan en la figura siguiente (Figura 5.25). Naturalmente no todas las relaciones están expresadas aquí, por dos motivos: obviedad (sólo hay que echar un vistazo a los nombres de los campos, que se corresponden) y claridad (si se pusieran todos los enlaces sólo se vería un amasijo de flechas).

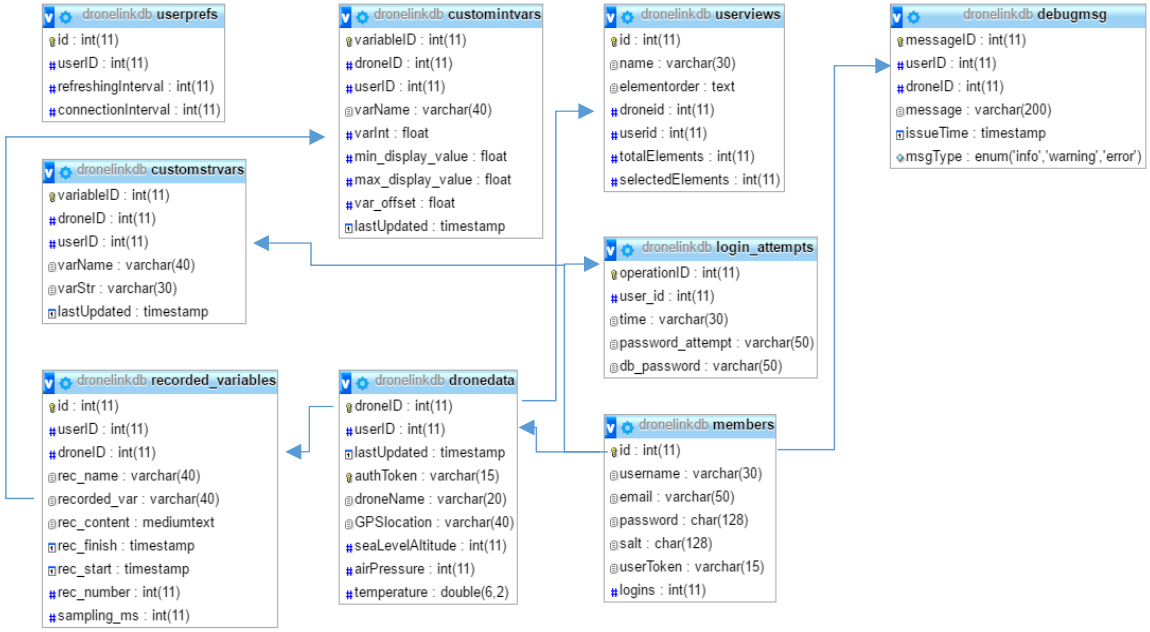


Figura 5.25: Tablas de la base de datos y sus relaciones.
Fuente: Elaboración propia.

5.4.1.3 El flujo de datos

En este apartado se exponen las operaciones efectuadas sobre la base de datos con mayor frecuencia y su implementación. Cuando el usuario o un dron solicitan efectuar alguna operación, esta se traduce en un conjunto de instrucciones que modifican los datos contenidos en la base. A modo de ejemplo, póngase el lector en la situación de que un usuario solicita eliminar una variable. Pues bien: el campo de la base de datos que se corresponde a ésta debe ser primero encontrado y posteriormente eliminado.

En el caso expuesto en el párrafo anterior, la eliminación es sencilla, pues ningún otro campo hace referencia a una variable. Sin embargo, al tratarse de eliminar un dron, por ejemplo, la cosa se complica: pues un dron "posee" variables, y estas, deben ser encontradas y eliminadas también. En conclusión: cualquier referencia al registro eliminado en cualquier otro campo, aunque esté contenido en otra tabla, debe ser eliminado también.

Esto es lo que entendemos por "flujo de datos": las manipulaciones necesarias sobre la base de datos para procesar cierta información. Nuevamente, y siguiendo en línea con la tónica de esta obra, no se expondrán todos y cada uno de los flujos de datos por motivos de extensión y objetivo del presente informe.

Es muy importante destacar y dejar claro que las operaciones que se exponen en los siguientes apartados incluyen diversas operaciones SQL de seguridad: si el usuario está autenticado, si tal variable pertenece a tal dron, si tal dron pertenece a tal usuario, etc. Estas consultas tampoco se incluyen por motivos de simplicidad.

5.4.1.3.1 Lectura de una variable

Las variables se almacenan en dos tablas separadas por motivos de velocidad en el acceso. Ya se pierde mucho tiempo utilizando un servidor SQL y no nos podemos permitir el lujo de desperdiciar más. Una tabla se encarga de las variables numéricas, mientras que la otra se encarga de las variables "*string*" o de cadena de texto.

Entonces, cuando se consulta una variable, el sistema pide a SQL que combine ambas tablas y que busque dónde está presente dicha variable y obtenga su valor. Al estar el campo definido como "cadena de texto" o "precisión de punto flotante", no es necesario declarar un tipo ambiguo como "*varchar*" (que admite ambos tipos) y que a la larga perjudica gravemente al desempeño de la base de datos.

Cabe destacar que en la rama de desarrollo de la plataforma que está enfocada a drones, existe una complicación adicional: pues hay variables "fijas" que se consideran comunes a todos los drones y se crean en la mismísima tabla que controla los drones que existen y los asocia al usuario. Estas variables son: *GPSlocation, temperature, airpressure y sealevelaltitude.* Existe una consulta SQL adicional que comprueba si la variable solicitada es una de estas 4 y las recupera del registro apropiado.

5.4.1.3.2 Eliminación de una variable

Este proceso es sencillo: se pide a SQL que localice la fila que se corresponde con la variable que se desea eliminar y se elimina de la base de datos. Si la variable fuera una de estas 4 especiales (véase el apdo. anterior), entonces se establece su valor a -9999 puesto que no pueden ser eliminadas de la base de datos. El sistema interpreta entonces que se desea ignorar dicha variable y no la devuelve cuando, por ejemplo, se piden todas las variables asociadas a un dron.

5.4.1.3.3 Creación de una variable

Cuando se pretende crear una variable el sistema primero comprueba que el nombre de dicha variable y el dron al que está asociada (nótese la "y") sean únicos. Esto significa que la situación mostrada en la Tabla 5.1, es absolutamente válida: puesto que, aunque el nombre es idéntico en dos casos, las variables pertenecen a dos drones distintos.

Tabla 5.1 Ejemplo de situación válida en una tabla de variables.

ID	varName	droneID	…
34	objetosRecogidos	52	…
…	…	…	…
83	objetosRecogidos	57	…

FUENTE: Elaboración propia.

En ningún caso se permiten los nombres de variables repetidos y asociados al mismo dron. El sistema devuelve un error y no escribe nada en la base de datos. Esto implica, como el lector ya habrá deducido, que es necesario revisar todas y cada una de las variables asociadas a un dron antes de insertar un registro nuevo.

Los nombres: "*seaLevelAltitude*", "*temperature*", "*airpressure*" y "*GPSlocation*" no están permitidos si las variables por defecto no han sido eliminadas. De volver a crearse, se crean como variables estándar.

5.4.1.3.4 Eliminación de un dron

Tratamos directamente la eliminación de un dron puesto que puede establecerse una analogía con las variables para su creación y lectura.

Cuando se elimina un dron, se elimina la fila de la tabla "drones" que se corresponde con el dron solicitado. A continuación, se eliminan todas las variables que referencien a ese dron de las tablas de variables. Finalmente, se recorre la tabla "vistas" y se eliminan también los registros que hagan referencia al dron.

5.4.1.3.5 Creación de un usuario

Cuando se registra un usuario al sistema se le asigna un identificador único de usuario, se encripta localmente (mediante JS) la contraseña, de modo que nunca abandone el ordenador del usuario sin cifrar, y se manda al servidor la solicitud después de comprobar que todos los datos son correctos.

El servidor siempre efectúa una comprobación de los datos por seguridad, pero primero se hace en el ordenador del cliente mediante JS para denegarla si es incorrecta y evitar, de antemano, mandar solicitudes incorrectas al servidor que ya sabemos que serán rechazadas.

Una solicitud incorrecta es cuando se da cualquiera de las siguientes condiciones sin prejuicio de otras circunstancias adicionales que puedan concurrir:

- No se han completado los campos obligatorios
- La contraseña es igual que el nombre de usuario o no cumple los requisitos de seguridad
- El correo electrónico no es correcto
- El nombre de usuario contiene caracteres inválidos
- Se ha introducido código SQL en alguno de los campos (inyección SQL)
- Se ha introducido código HTML en alguno de los campos

5.4.2 La plataforma

La plataforma es una solución que cumple todas las características y requisitos expuestos a lo largo de los capítulos anteriores de esta obra. La siguiente figura muestra la página de inicio de sesión en la plataforma, que es lo primero que ve el usuario cuando accede al software.

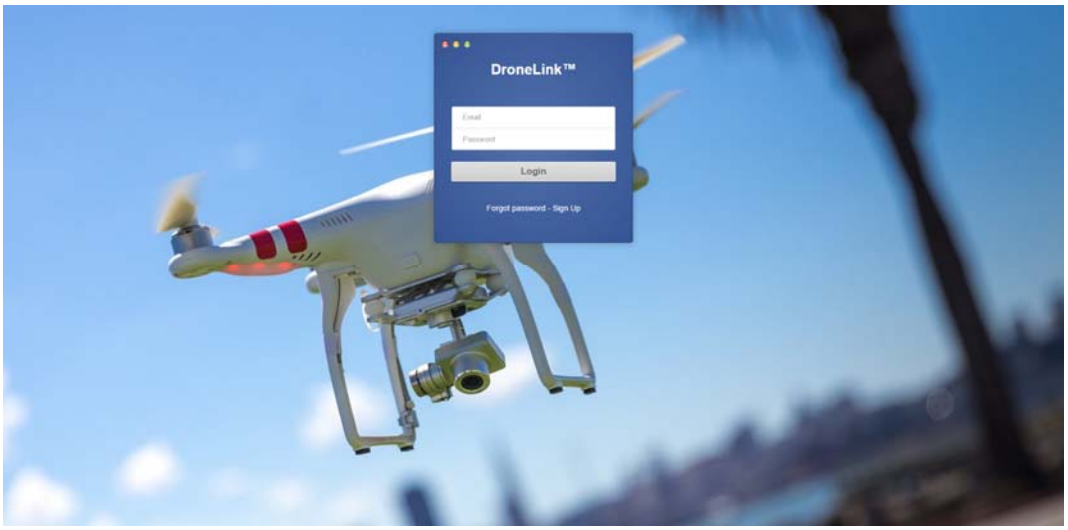


Figura 5.26: Pantalla de inicio de sesión en el software.
Fuente: Elaboración propia (contiene imágenes de licencia CC).

5.4.2.1 Descripción general

La plataforma está pensada para poder ser usada fácil e intuitivamente incluso por usuarios principiantes en informática. Cuenta con un menú de navegación en la parte izquierda para acceder a las funcionalidades principales, una barra superior de estado: que monitoriza el estado actual de los drones, y un panel principal de inicio donde también se puede acceder a las funcionalidades de la misma.

Todo el sistema está basado en un concepto de redundancia: la misma operación se puede efectuar desde diversos lugares y mediante diversos procedimientos. No hay una sola forma de hacer las cosas en DroneLink™. Por ejemplo: cada vez que aparece el nombre de un dron en la plataforma, lo hace con un hipervínculo a la página de visualización del dron en cuestión.

Existen dos ramas de desarrollo de la plataforma: la genérica (sin nombre comercial en el momento de la redacción del presente documento) y la específica para drones (DroneLink™). Esta segunda, derivada de la primera, se enfoca exclusivamente a drones y provee herramientas tales como: mapas con seguimiento GPS y monitorización de batería.

El motivo de la separación y diferenciación de las versiones es la voluntad de extensión del proyecto a múltiples campos de aplicación sin restar especificidad al mismo.

5.4.2.1.1 Funcionamiento y ejemplo práctico

En este apartado se expondrá un ejemplo práctico de uso de la plataforma con un "dron" de ejemplo basado en la plataforma Arduino™. No se ha utilizado un dron real basado en Arduino™, sino una maqueta que incluye sólo la lógica del dron, pero sin actuadores (hélices, timones, etc…).

Para ello primero debemos registrarnos en el sistema. Hemos introducido, como credenciales, "test@test.com" de correo electrónico, "testA1" de nombre de usuario y "A1test" de contraseña (todo sin comillas).

Una vez registrados, iniciamos sesión en la pantalla inicial. Nos encontramos con el panel de control donde se nos informa de que no tenemos drones disponibles en nuestra cuenta, tal y como se aprecia en la siguiente figura (Figura 5.27):

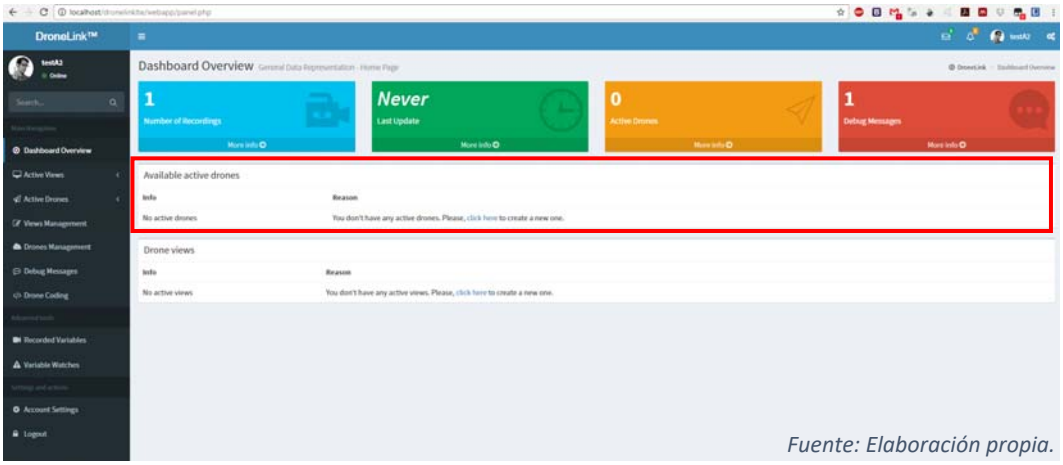


Figura 5.27: Captura de pantalla donde la plataforma indica que no hay drones registrados.

Parece evidente, pues, que el primer paso a seguir es registrar un nuevo dispositivo en la plataforma. Esto podríamos hacerlo desde el menú lateral izquierdo, pero la plataforma nos ofrece un enlace en el propio mensaje, ofreciéndonos crear uno. Pulsamos sobre el enlace. Esto nos lleva a una pantalla tal como la de la Figura 5.28:

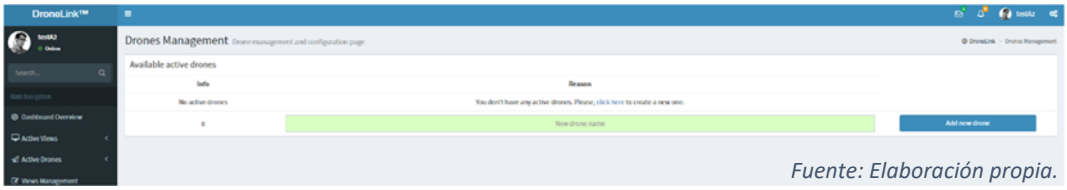


Figura 5.28: Recorte de pantalla donde la plataforma ofrece crear un nuevo dron.

Aquí introducimos un nombre para el dron a nuestra libre elección. Por ejemplo, lo nombraremos "Dron de Prueba". Este campo es útil para identificarlos cuando se dispone de varios de ellos funcionando simultáneamente.

Una vez el dron está creado, este aparece en el menú lateral, y, sin salir de la misma página, podemos añadir variables desde la nueva tabla que ha aparecido más abajo.

Cómo ya se ha mencionado anteriormente, hay variables que vienen por defecto al crear un nuevo dron y son añadidas automáticamente. Pueden ser eliminadas si se desea pulsando sobre la cruz que aparece a la derecha de la pantalla, en la fila de cada variable.

Pulsando sobre el icono con forma de ojo podemos ver un gráfico con el estado de esa variable en tiempo real. Si pulsamos sobre el lápiz, podremos editarla. El resultado obtenido es el que se muestra en la Figura 5.29:

Fuente: Elaboración propia.

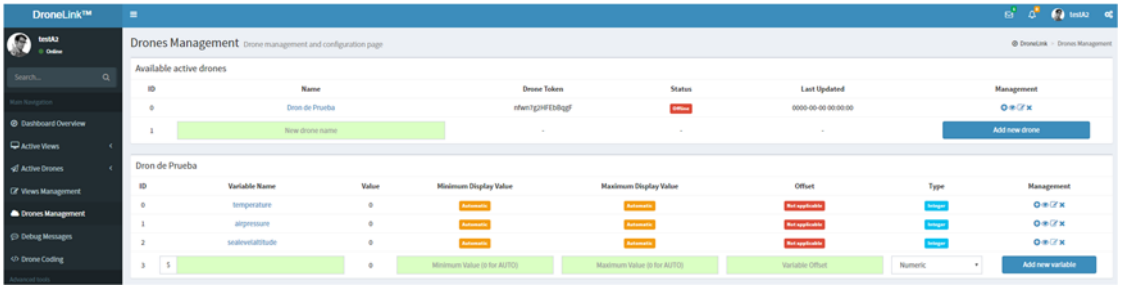


Figura 5.29: Recorte de pantalla que muestra el dron de prueba ya creado y sus variables por defecto.

En esta página podemos apreciar otra información muy importante. Cabe que anotemos el código que se muestra en la columna "Drone Token" correspondiente a la fila del dron recién creado. Hay que respetar mayúsculas y minúsculas.

Ahora, si efectuamos una conexión Bluetooth™ y ejecutamos la utilidad DroneLink Bridge™ suministrada junto a la plataforma en el ordenador receptor de la señal del dron, podemos establecer la variable "temperature" a 20 utilizando un comando tal como este:

```
SerialBT.println("nfwn7g2HFEbBqgF, temperature, 20")
```

Se pone el identificador, coma, el nombre la variable, coma, y el valor. Los cambios serán inmediatamente reflejados en tiempo real y veremos el cambio reflejado en el gráfico de tiempo real y en el valor numérico de la variable. En la Figura 5.30 podemos ver una parte del panel de control donde se muestran los gráficos de todas las variables simultáneamente.

Fuente: Elaboración propia.

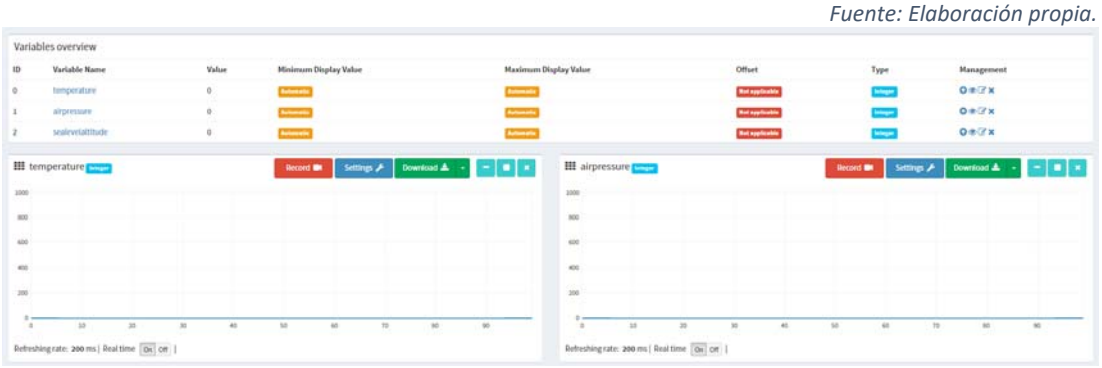


Figura 5.30: Recorte de pantalla de la sección "drones", viendo el dron de prueba.

Ejecutamos la instrucción en el dron de prueba, en el dispositivo arduino, realizando el montaje DroneLink Bridge™ y lo que obtenemos es lo que se muestra en la Figura 5.31: un salto en el valor de la variable de 0 a 20. Se aprecia bastante pequeño porque la escala está configurada, por defecto, de 0 a 1024. Esto se puede cambiar para cada variable por separado en el momento de crearlas, en la

pantalla de gestión. Puede ser cambiado posteriormente en la misma pantalla. Incluso se puede establecer la escla en modo automático para que se ajuste dinámicamente al rango de valores de entrada en tiempo real.

Fuente: Elaboración propia.

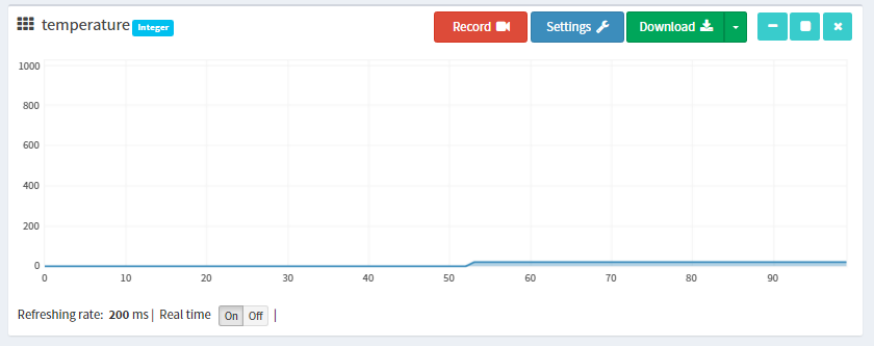


Figura 5.31: Recorte de pantalla donde se ve el cambio de "0" a "20" de la variable "temperature".

5.4.2.1.2 Relación de archivos de código del servidor

Los archivos de código de lado de servidor (*.php y *.inc.php) que se han escrito son los siguientes (se indica entre paréntesis el número de líneas de código que se han escrito para cada uno de ellos):

Directorio Includes (librerías incluibles en otros archivos)

- includes/content_manager.php (624)
- includes/db_connect.php (3)
- includes/functions.php (142)
- includes/login_process.php (21)
- includes/logout.php (21)
- includes/notifications_manager.php (79)
- includes/psl_config.php (13)
- includes/register.inc.php (92)
- includes/register.php (59)
- includes/register_success.php (12)

Directorio *webapp* (aplicación principal y API)

- webapp/datalink.php (86)
- webapp/datalog.php (46)
- webapp/datarec.php (65)
- webapp/dronesmanage.php (523)
- webapp/droneview.php (592)
- webapp/messages.php (392)

- webapp/msgread.php (63)
- webapp/panel.php (464)
- webapp/recget.php (64)
- webapp/recordedvars.php (310)
- webapp/variableview.php (472)

Directorio raíz

- error.php (19)
- index.php (12)
- login.php (41)

Estos archivos son los que conforman la aplicación. Todos y cada uno de ellos son piezas imprescindibles que se entrelazan a la perfección, dando lugar a la plataforma DroneLink™. En total son 4265 líneas de código.

5.4.2.2 Sistema de inicio de sesión seguro

El sistema de inicio de sesión seguro proporciona un método para iniciar y cerrar sesión con total seguridad y protección frente a las principales amenazas. Posee una protección especial que bloquea durante un tiempo al usuario que introduzca incorrectamente su contraseña en 3 ocasiones. Todos los intentos de sesión fallidos quedan registrados en el sistema.

5.4.2.3 Clases de manipulación de datos

Para evitar repetir código por todas partes, se han creado unos archivos especiales que contienen lo que en informática se denomina POO (OOP en inglés), que significa "programación orientada a objetos". Estos archivos se auto-instancian de manera que la clase no es estática, pero sí existe una sola instancia de la misma. Con esta técnica se consigue abstraer de manera muy significativa la programación del sistema y, sin llegar ser un MVC (*model, view, controller*) o modelo vista controlador, constituye una forma muy elegante y simple de hacer las cosas.

El archivo más utilizado en todo el programa y que precisamente implementa este modelo es el *"content_manager.php"*.

5.4.2.4 El panel de control

El panel de control está basado en AdminLTE™ que es una plantilla bajo licencia de libre modificación, distribución y uso comercial. Cuenta con un menú lateral izquierdo con las principales opciones, una barra de estado superior desde donde se pueden consultar los mensajes que mandan los dispositivos registrados y las notificaciones (consejos) que nos da la plataforma en función del uso que hacemos de ella. La siguiente figura (Figura 5.32) muestra la vista principal del panel de control.

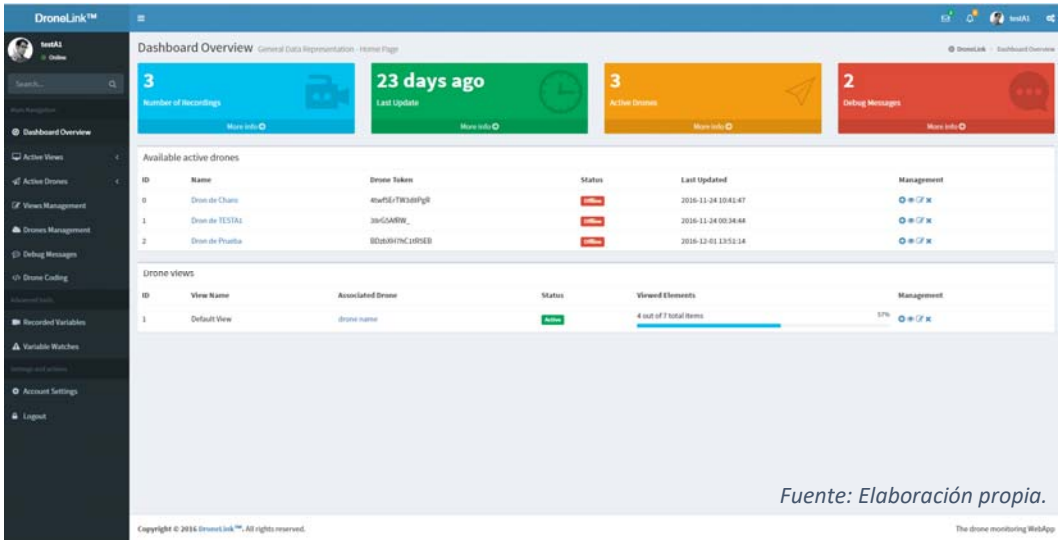


Figura 5.32: Vista principal del panel de control, se ven 3 drones y una "vista" creada.

Las distintas funciones que implementa la plataforma DroneLink™ son:

- Vista general
- Vistas activas
- Drones activos
- Gestión de vistas
- Gestión de drones
- Mensajes de depuración
- Programación de drones
- Variables grabadas
- Alertas en variables
- Opciones generales

Estas funciones son accesibles principalmente, aunque no exclusivamente, desde el menú lateral izquierdo (véase la Figura 5.33):

Fuente: Elaboración propia.

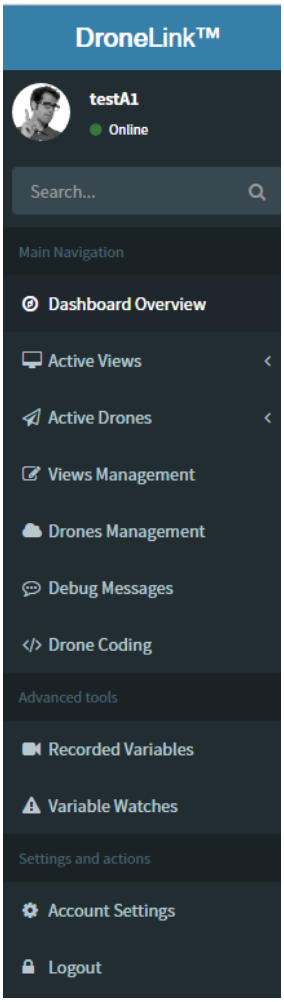


Figura 5.33: Menú lateral izquierdo de la plataforma.

En el siguiente apartado (5.4.2.4.1) se describen estas funciones con mayor detalle y se incluyen capturas de pantalla que ilustran lo expuesto. Las categorías que poseen una flecha al lado son desplegables: situando el cursor sobre ellas se despliega un menú que incluye subcategorías de las mismas. Por ejemplo, en la figura anterior (Figura 5.33) al situarnos sobre “active drones” nos aparece un menú con todos los drones asociados a nuestra cuenta.

5.4.2.4.1 Funcionalidades principales

5.4.2.4.1.1 Vista general

La vista general muestra la información más relevante sobre el estado del sistema y, aún más importante, enlaces rápidos a los detalles sobre la misma. Esta vista es la que podemos ver en la Figura 5.32. En particular muestra el número de grabaciones que ha realizado el usuario, la última vez que se actualizó un dron (en texto), el número de drones activos, el número de mensajes de depuración recibidos, los drones existentes en una tabla e información relacionada y otra tabla con las vistas existentes.

5.4.2.4.1.2 Gestión de drones

Esta página permite crear, modificar, y eliminar drones. Cuando se crea un dron se genera automáticamente un código que le identifica y aparece en esta página también. Desde aquí, como el título indica, se pueden gestionar los drones, y las variables que dependen de ellos.

Para cada dron creado desde la primera tabla, la de drones, aparecerá anexionada en la parte de abajo del documento una tabla nueva que contendrá todas las variables de ese dron. Desde esta tabla se pueden añadir y eliminar variables a voluntad.

Aquí es posible, entre otras cosas, decidir qué límites se van a mostrar en el gráfico de cada variable (mínimo y máximo) y qué *offset* (o desviación) se va a aplicar a cada variable también. El *offset* no es más que un valor que se suma algebraicamente (si es negativo: se resta) al valor leído antes de ser mostrado en el gráfico.

En la siguiente figura se ve la página de gestión de drones (Figura 5.34):

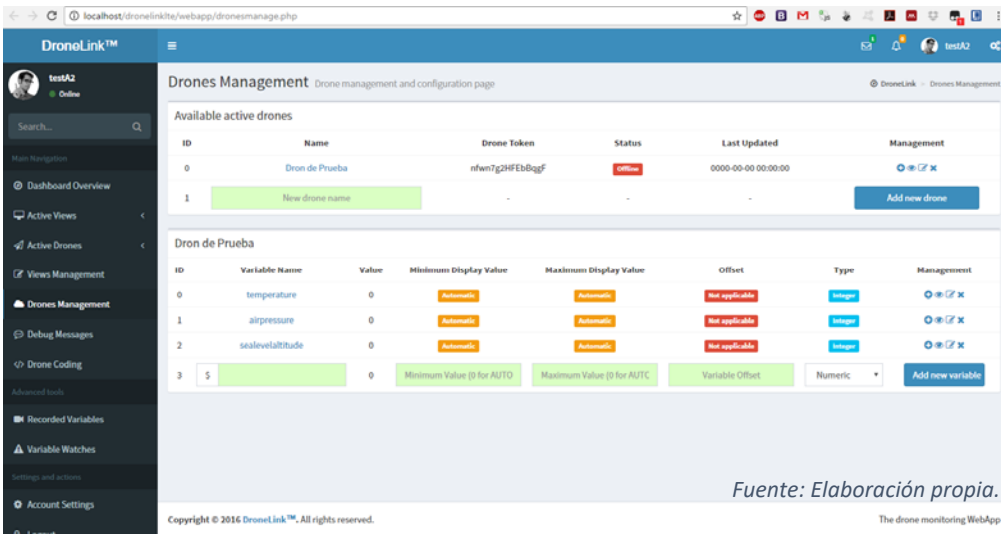


Figura 5.34: Página para la gestión de drones y variables.

5.4.2.4.1.3 Grabación de datos y exportación

Las variables pueden grabarse desde el panel de "visualización de un dron" o el de "visualización de una vista" presionando el botón rojo "grabar". Al pulsar "detener", saldrá un cuadro de diálogo preguntándonos el nombre de la grabación que acabamos de realizar (Figura 5.35).

Fuente: Elaboración propia.

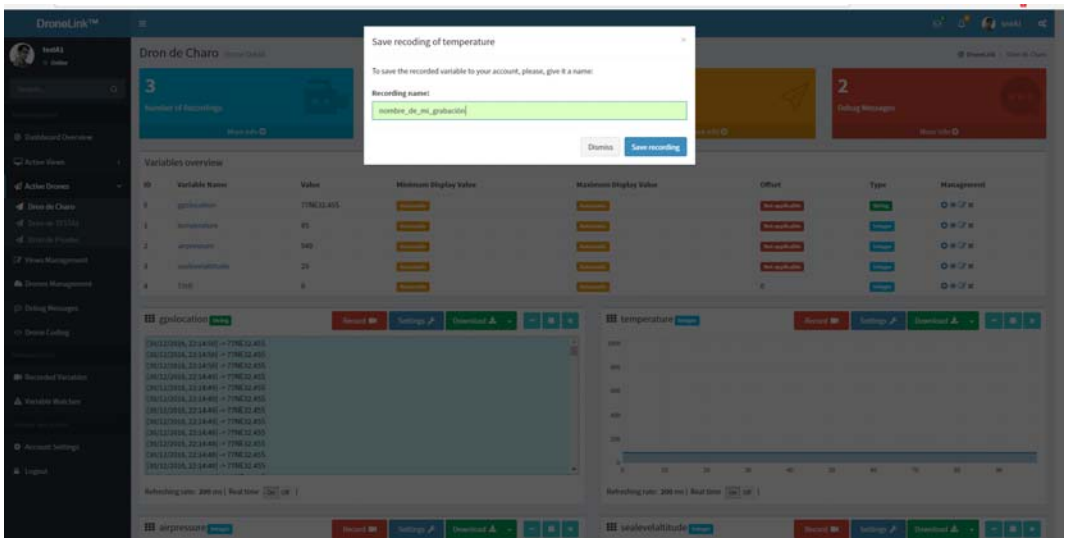
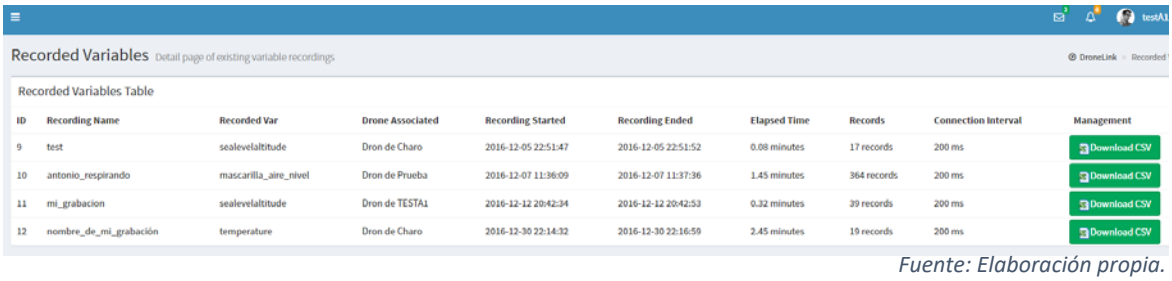


Figura 5.35: Cuadro de diálogo para guardar una grabación.

Al hacer clic sobre "save recording" (guardar grabación), automáticamente se registrará en el sistema la grabación que acabamos de realizar bajo ese nombre, y aparecerá en el apartado "Variables grabadas" (Figura 5.36).



ID	Recording Name	Recorded Var	Drone Associated	Recording Started	Recording Ended	Elapsed Time	Records	Connection Interval	Management
9	test	sealevelaltitude	Dron de Charo	2016-12-05 22:51:47	2016-12-05 22:51:52	0.08 minutes	17 records	200 ms	Download CSV
10	antonio_respirando	mascarilla_aire_nivel	Dron de Prueba	2016-12-07 11:36:09	2016-12-07 11:37:36	1.45 minutes	364 records	200 ms	Download CSV
11	mi_grabacion	sealevelaltitude	Dron de TESTA1	2016-12-12 20:42:34	2016-12-12 20:42:53	0.32 minutes	39 records	200 ms	Download CSV
12	nombre_de_mi_grabación	temperature	Dron de Charo	2016-12-30 22:14:32	2016-12-30 22:16:59	2.45 minutes	19 records	200 ms	Download CSV

Fuente: Elaboración propia.

Figura 5.36: Lista con las grabaciones realizadas.

La realización de grabaciones se puede encontrar en el apartado "Variables grabadas" y desde allí podrá descargarse la grabación en formato CSV (*Coma separated value*) o valores separados por coma; este fichero puede ser visualizado en un programa de hojas de cálculo como Microsoft® Excel® para su posterior tratamiento.

Esta página también muestra información adicional como la duración de la grabación, la frecuencia de actualización, y el número de registros.

5.4.2.4.1.4 Visualización de variables

Las variables se pueden visualizar accediendo desde el menú izquierdo (Figura 5.37) al dron en cuestión (el que posee la variable) o desde alguna "vista" que hayamos configurado previamente y que también incluya a la variable que deseemos consultar en tiempo real.

Fuente: Elaboración propia.

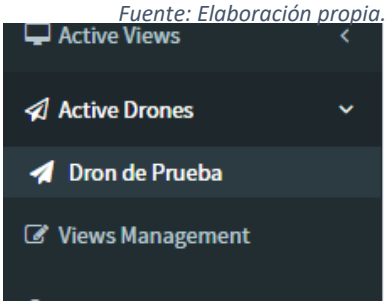


Figura 5.37: Menú desplegable que contiene los drones que hemos registrado en el sistema.

En caso de seleccionar un dron, disponemos de las siguientes opciones: visualizar la variable, detener la visualización en tiempo real, cambiar los ejes de los gráficos (la escala), reanudar la visualización en tiempo real, descargar los valores que muestra el gráfico actualmente, ampliar la variable a pantalla completa, arrastrar con el ratón las ventanitas que contienen las variables y reorganizarlas a nuestro gusto, iniciar grabaciones simultáneas de variables y detenerlas.

Fuente: Elaboración propia.

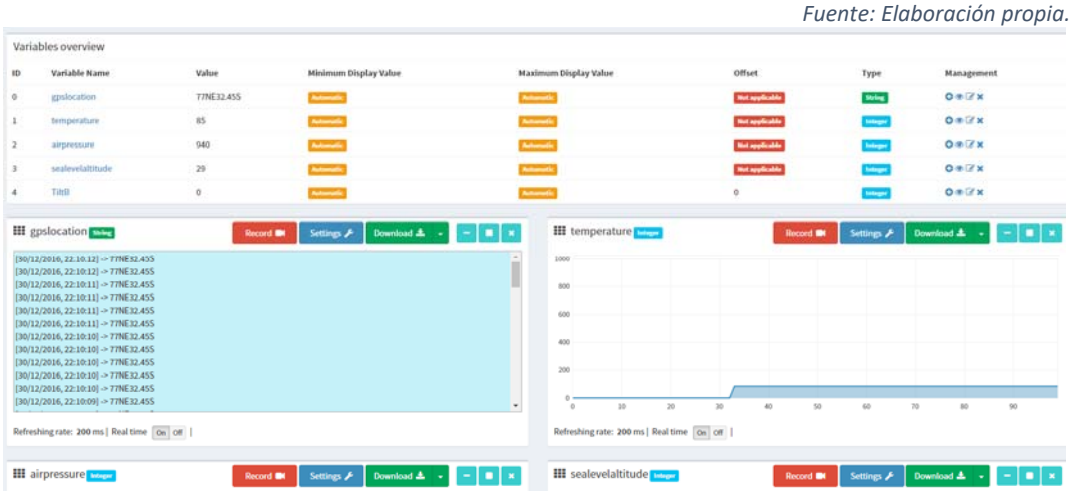


Figura 5.38: Vista de las variables que contiene un dron.

Como se ve en la Figura 5.38, las variables de texto se actualizan en una caja de texto y se van anexionando líneas junto con la fecha y hora de la recepción. Por el contrario, las variables numéricas se actualizan en un gráfico de ejes personalizables.

5.4.2.5 Pruebas en diferentes sistemas operativos y navegadores

El sistema se ha probado satisfactoriamente en las siguientes plataformas y sistemas operativos mostrados en la tabla siguiente (Tabla 5.2):

Tabla 5.2: Sistemas operativos y navegadores compatibles con el sistema.

	IE[6]®/Edge®	Chrome™	Safari®	Firefox™	Opera™
Windows® XP® SP3[7]	✓	✓	✓	✓	✓
Windows Vista/7	✓	✓	✓	✓	✓
Windows 8/10	✓	✓	✓	✓	✓
Ubuntu™ 16.04 LTS (Linux)	✗	✓	✗	✓	✗

FUENTE: Elaboración propia.

Como vemos el software es compatible en los principales navegadores y sistemas operativos siempre y cuando utilicemos las últimas versiones disponibles del software y, en el caso de Windows®, el último *service pack* (SP)[8].

5.4.3 El software de enlace y depuración local

Para enlazar el dron con la plataforma DroneLink™ disponemos de varias opciones, a saber: XBee® (punto de acceso intermedio), Bluetooth™ (punto de acceso intermedio), puerto serie (punto de acceso intermedio), Ethernet (directo), o WiFi (directo).

Como ya se ha explicado en el apartado 5.2.3 (y se ve en la Figura 5.9), hay tecnologías que requieren de un ordenador intermedio que decodifique la señal y la traduzca al estándar TCP/IP sobre la red, para efectuar la conexión con el servidor DroneLink™. Esa tarea justamente es la del software complemento de enlace DroneLink Bridge™, que proporciona un método sencillo y directo para las tecnologías XBee®, Bluetooth™ y puerto serie.

Este software se conecta a un puerto local del ordenador intermedio (y es allí donde debe ejecutarse) y crea las cadenas de conexión al servidor oportunas. A continuación, las manda a la dirección URL (dirección de internet) o a la IP especificada.

[6] IE®: Internet Explorer®. Debido a su similaridad con IE® (mismo motor), se consideran el mismo navegador a efectos prácticos.

[7] En Windows® XP® se ha utilizado la versión 8 de IE®. La versión 7 no es compatible. De Chrome se ha utilizado la 50, que es la última compatible con esta versión de Windows®.

[8] Un SP, o *service pack*, es una gran actualización del sistema operativo que incluye numerosas novedades y correcciones de errores.

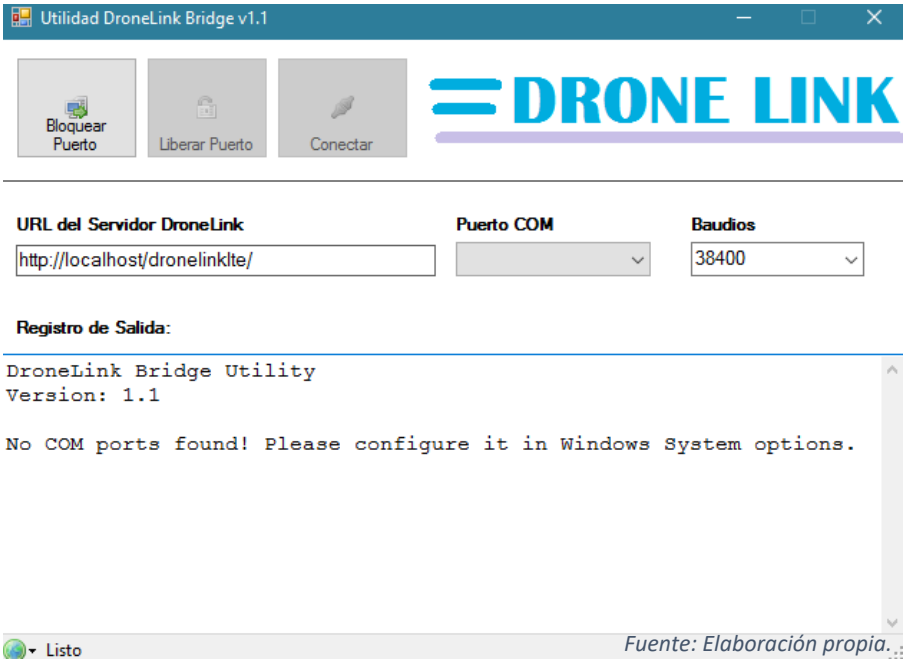
La siguiente tabla (Tabla 5.3) muestra los modos de conectar el dron a la red según la tecnología de comunicación que utilicemos, y sus características:

Tabla 5.3: Características y modos de conexión de las distintas tecnologías soportadas.

	Xbee®	Puerto Serie	Bluetooth™	Ethernet	WiFi
Conexión directa	✗	✗	✗	✓	✓
Bajo consumo	✓	✓	✗	✓	✗
Inalámbrico	✓	✗	✓	✗	✓

FUENTE: Elaboración propia.

El software DroneLink Bridge™ tiene un solo formulario, desde donde se pueden introducir todos los ajustes necesarios. En la siguiente figura (Figura 5.39), se puede ver el aspecto que presenta:



Fuente: Elaboración propia.

Figura 5.39: Utilidad de conexión DroneLink Bridge™

La utilidad está disponible en inglés y español, a diferencia de la plataforma DroneLink™, que, de momento, sólo está disponible en inglés. Esta utilidad efectúa el enlace entre el dron y la plataforma DroneLink™. Está disponible para sistemas operativos Windows® (.NET Framework™) y Linux (utilizando Mono™).

La suite DroneLink™ también incluye software de depuración local "Serial Plotter" compatible con el formato de conexión con el que trabaja DroneLink™ Bridge para una rápida depuración local. Si se instala un servidor local como XAMPP, y sobre éste se instala DroneLink™ y DroneLink™ Bridge el propio ordenador trabaja de servidor y se pueden llevar a cabo tareas de depuración local en un entorno avanzado.

5.4.3.1 Biblioteca C para Arduino™ sobre TCP/IP ethernet

Esta biblioteca ha sido desarrollada para conectarse a DroneLink™ sobre *ethernet* de una manera muy sencilla utilizando un Arduino™ que cuente con un módulo ethernet. No necesita ningún servidor "proxy" o intermediario (como DroneLink™ Brigde), puesto que la conexión a internet es directa.

Sencillamente hay que incluir la biblioteca en el código de Arduino™, llamar a un método de configuración de la conexión y ejecutar una sencilla instrucción cada vez que se quiera actualizar la variable. La siguiente figura (Figura 5.40) muestra la sencillez de uso de la librería C para Arduino™ en el IDE (*integrated development environment*) oficial:

Fuente: Elaboración propia.

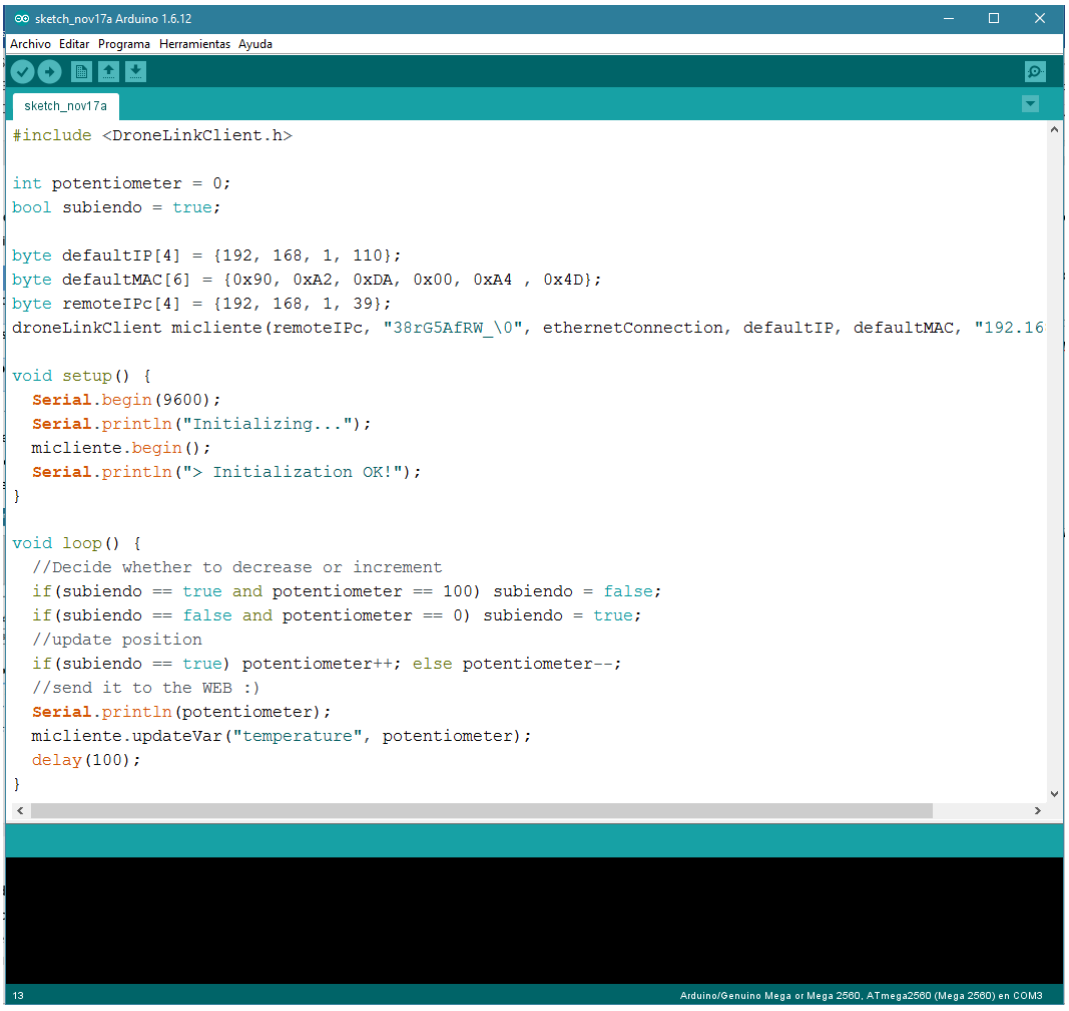


Figura 5.40: Se muestra el IDE de Arduino™ con un código que genera una onda triangular y la manda a DroneLink™

5.4.3.2 Biblioteca C para Arduino™ sobre Bluetooth™ y XBEE®

Las bibliotecas necesarias ya van incluidas dentro de la utilidad DroneLink™ Bridge, por lo que no es necesario utilizar bibliotecas adicionales en el propio Arduino™. Solamente hay que mandar por puerto serie una cadena que sea del formato: "token del dron, nombre de la variable, valor". De este modo el ordenador intermedio ejecutando DroneLink™ Bridge hará las conversiones pertinentes y mandará la petición al servidor DroneLink™.

5.4.3.3 Software Serial Plotter para PC

El software Serial Plotter es una utilidad desarrollada para interpretar cadenas del formato descrito en el apartado 5.4.3.2. Muestra la evolución de los valores por la pantalla del ordenador en tiempo real y guarda todo lo que muestra en un archivo CSV compatible con programas de hoja de cálculo tales como Microsoft® Excel® o OpenOffice™ Calc.

5.4.3.4 API genérica

En caso de utilizar un dispositivo genérico para el que no haya librería disponible, o usar una tecnología que permita conexión directa a internet, la forma más fácil de comunicarse con DroneLink™ es utilizar su API. Consta de dos métodos: uno para obtener valores y otro para guardarlos.

Para obtener todos los valores asociados al dron basta con formar una petición de la siguiente forma (sustituyendo la negrita por los valores pertinentes):

https://server/webapp/datalink.php?droneToken=**393r9wGDI**&userToken=**D32kWo3k_**

Para guardar un valor debe formarse una petición de la esta otra manera (sustituyendo la negrita por los valores pertinentes):

https://server/webapp/datalog.php?authToken=**393r9wGDI**&cathegory=**GPS**&value=**45NE25NW**

Los códigos necesarios pueden encontrarse iniciando sesión con el nombre de usuario y contraseña y consultando el apartado gestión de drones "drones management".

5.4.4 El prototipo de pruebas

El prototipo que se ha desarrollado cuenta con un módulo GPS, un sensor de altitud, rotación y temperatura y un actuador. El sistema trabaja sobre XBEE® y se conecta a DroneLink™ mediante DroneLink™ Bridge. El prototipo simula ser un dron y actualiza todas las variables en tiempo real.

El prototipo incluye un módulo de GPS con antena, un sensor de altitud, temperatura, humedad, giro y aceleraciones (BMP 280) , dos leds de estado, un módulo Bluetooth HC-05 de Bluetooth™ y un XBEE®.

La siguiente figura (Figura 5.41) muestra la configuración del mismo:

Fuente: Elaboración propia.

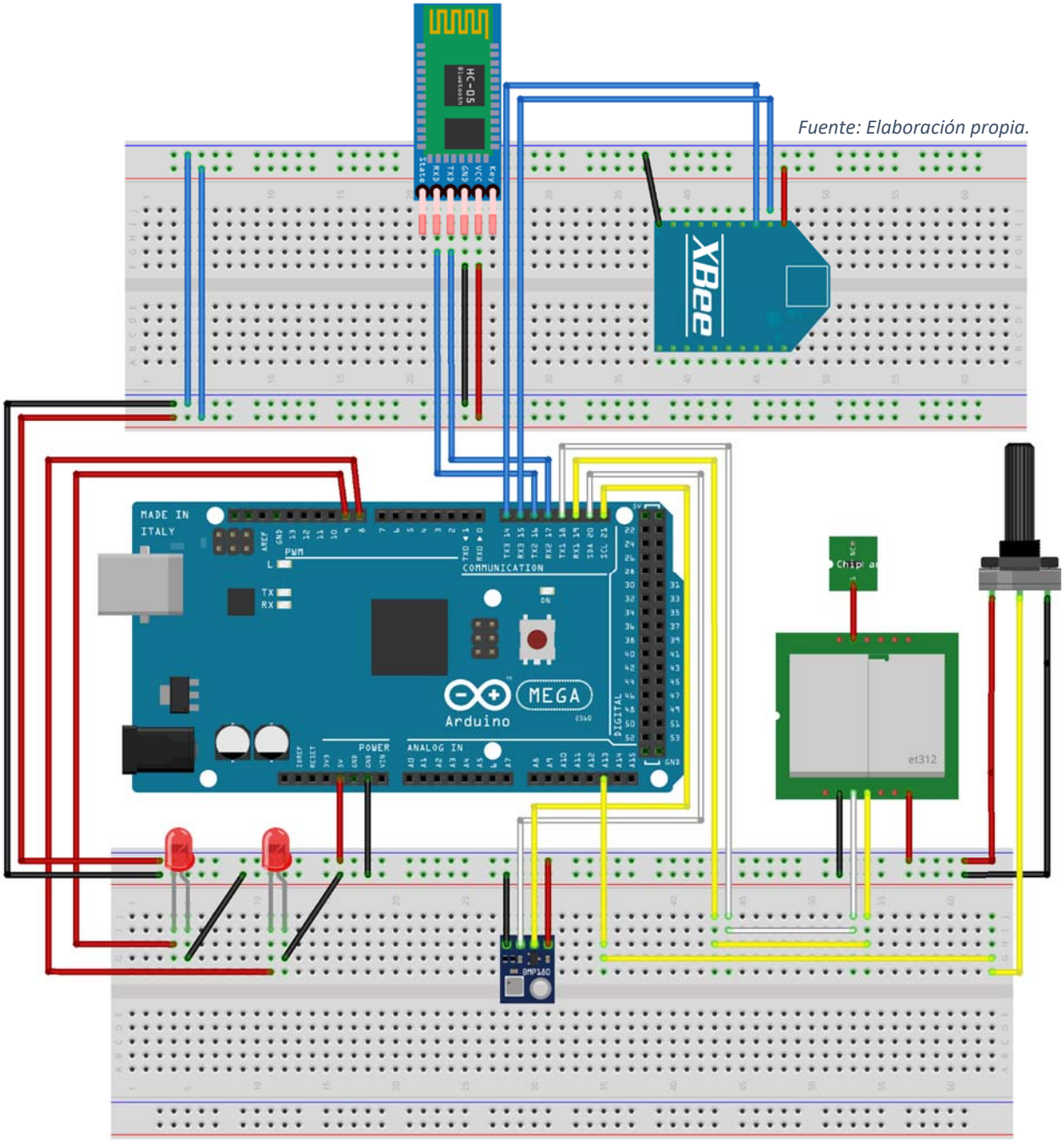


Figura 5.41: Dibujo esquemático del modelo construido para realizar pruebas.

6 RESUMEN DE RESULTADOS, PRESUPUESTO E IMPACTO MEDIOAMBIENTAL

6.1 RESUMEN DEL PRESUPUESTO

Debido al carácter fuertemente informático de este proyecto, y a que se han utilizado herramientas de código abierto, el presupuesto para el mismo no asciende demasiado, tal y como se puede apreciar en la siguiente tabla (Tabla 6.1):

Tabla 6.1: Presupuesto para la realización del presente TFG.

Ref.	Concepto	Área	Cant.	Importe	Monto
1	Arduino Pro Micro™	Prototipo	1	9,90 €	9,90 €
2	Ethernet Shield	Prototipo	1	6,36 €	6,36 €
3	Arduino UNO™	Prototipo	1	26,49 €	26,49 €
4	Módulo XBEE™	Prototipo	2	34,35 €	68,70 €
5	Cableado	General	1	2,54 €	2,54 €
	IVA: 21%		**Subtotal (sin IVA)**		**90,05 €**
			Importe IVA		23,94 €
			Total (IVA inc.)		**113,99 €**

FUENTE: Elaboración propia.

El presupuesto de implementación del proyecto en un entorno de producción depende enormemente de la capacidad del servidor, del volumen de peticiones, del sistema operativo utilizado, y de un sinfín de particularidades más.

A continuación (Tabla 6.2), se expone el presupuesto anual de una posible implementación típica, para monitorizar un dron en modo local (el ordenador actúa de servidor, si se conecta a internet, los datos se vuelven accesibles desde cualquier parte del mundo).

Tabla 6.2: Presupuesto de implementación típico, en local.

Ref.	Concepto	Cant.	Importe	Monto (IVA inc.)
1	Arduino UNO™	1	9,90 €	9,90 €
2	Módulo XBEE™	1	34,35 €	34,35 €
3	Portátil Medion E4213	1	199,00 €	199,00 €
4	Licencia de uso de DroneLink™	1	-	-
	IVA: 21%	Subtotal (sin IVA)		**192,17 €**
		Importe IVA		51,08 €
		Total (IVA inc.)		**243,25 €**

FUENTE: Elaboración propia.

Si se desea implementar en un servidor de pago profesional, en vez de en un portátil, y cobrar a los clientes finales una tarifa para que accedan al producto sencillamente con su nombre de usuario y contraseña, el presupuesto podría ser el siguiente (Tabla 6.3):

Tabla 6.3: Presupuesto de instalación en servidor de internet profesional ARSYS®.

Ref.	Concepto	Cant.	Importe	Monto
1	Alojamiento Web Ilimitado (ARSYS®)	1	179,40 €	179,40 €
2	Servicios SSL	1	12,99 €	12,99 €
3	Soporte técnico (incluido en ref. 1)	1	- €	- €
4	Mantenimiento del Servidor (inc. ref. 1)	1	- €	- €
	IVA: 21%	Subtotal (sin IVA)		**151,99 €**
		Importe IVA		40,40 €
		Total (IVA inc.)		**192,39 €**

FUENTE: Elaboración propia. NOTA: Esta tabla hace referencia a costes anuales.

Finalmente cabe destacar que estos presupuestos incluyen únicamente los costes derivados de la plataforma DroneLink™ y/o el sistema de cableado: en ningún caso se incluyen los costes derivados del dispositivo y/o dron que se vaya a conectar a la misma.

6.2 ANÁLISIS Y VALORACIÓN DEL IMPACTO MEDIOAMBIENTAL

El impacto medioambiental de este proyecto es el impacto que tenga la fabricación de la infraestructura necesaria para su implementación y el impacto derivado de la utilización de la misma.

Como que resulta evidente que no se puede analizar al detalle el coste medioambiental de fabricar un ordenador portátil (una de las infraestructuras que se utilizan) porque, de por sí sólo, constituiría otro trabajo de fin de grado distinto a este; se analizará el coste medioambiental de mantener en funcionamiento un servidor HTTP.

6.2.1 Coste medioambiental de mantenimiento de un servidor HTTP

Mantener un servidor HTTP estándar (se analiza el coste del servidor presupuestado en el apartado 6.1 (Tabla 6.3)) tiene un coste medioambiental fijo, por la corriente que consume que se puede considerar aproximadamente constante durante su funcionamiento. Luego, estos servidores están agrupados y en una sala, con personal trabajando, iluminación, agua caliente sanitaria, etc. Estos gastos, como veremos a continuación, no son nada despreciables y suponen la mitad del impacto total.

Estudiaremos el consumo eléctrico del centro de datos por ser éste el principal factor de impacto medioambiental.

Según una nota de prensa de ARSYS® (Arsys, 2012), el 50% del consumo eléctrico habitual en un centro de datos se destina a los servidores en sí. Un 35% se destina a climatización de salas técnicas y un 15% a la alimentación de infraestructuras (alumbrado, dispositivos de control, …). Dichas magnitudes se pueden apreciar con mayor claridad en la siguiente figura (Figura 6.1):

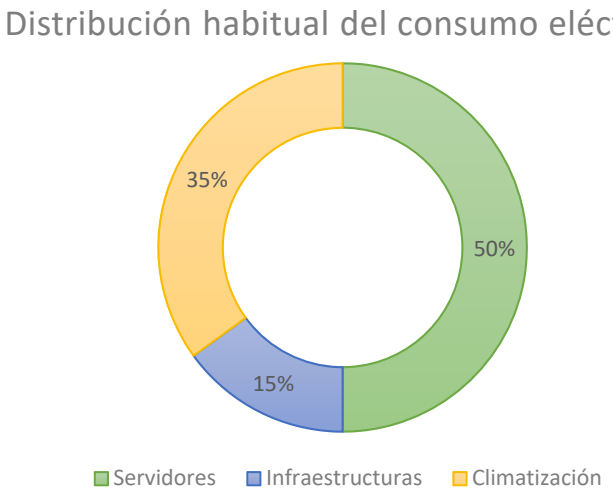


Figura 6.1: Distribución habitual del consumo eléctrico en centros de datos españoles. Fuente: Elaboración gráfica propia. Datos de ARSYS® S.L.U.

Afortunadamente, ARSYS® mejoró en el año 2012 sus infraestructuras consiguiendo destinar una mayor parte del consumo eléctrico a sus servidores y reduciendo el consumo eléctrico general, obteniendo una distribución como la que se muestra en la siguiente figura (Figura 6.2):

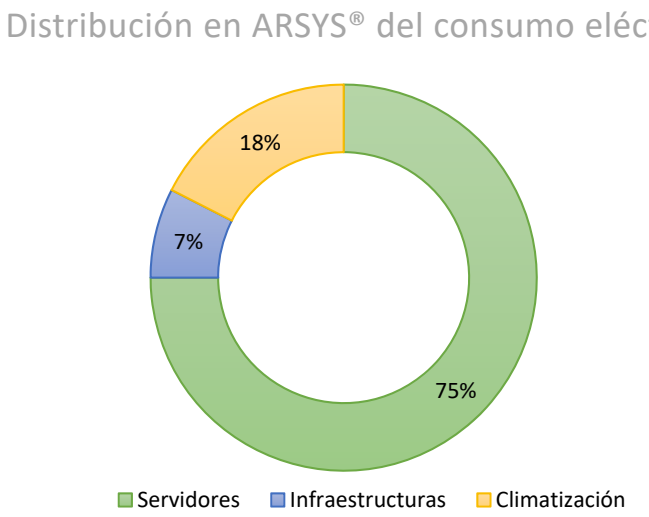


Figura 6.2: Distribución del consumo eléctrico tras la mejora de ARSYS®
Fuente: Elaboración gráfica propia. Datos de ARSYS® S.L.U.

Los datos que muestra la figura superior (Figura 6.2), han sido calculados a partir de la nota de prensa publicada por ARSYS®, donde especifica la mejora en términos de PUE (*Power Usage* Effectiveness) o eficacia del uso de la energía. Este factor hace referencia a cuántos kW son utilizados por los servidores de cada kW consumido. Así, un PUE de 2 (el habitual) significa que, de cada 2 kW consumidos, sólo 1 kW va a las instalaciones de servidores (véase la Figura 6.1). Un PUE de 1 es el límite inferior y el mejor resultado posible: significando que toda la potencia va a los servidores.

El PUE de ARSYS®, después de los cambios, ha sido reducido a 1,35 (por cada kW que consumen los servidores, sólo 0,35 van a las instalaciones auxiliares); consiguiendo una mejora del 33% sobre el PUE habitual en el sector.

El consumo de un servidor como los utilizados por ARSYS® podríamos estimarlo en la parte fija de nuestro servidor más un porcentaje de los gastos comunes en proporción directa con el número de servidores que posean. Si hay 100 servidores, el nuestro sería responsable de un 1% de los costes medioambientales de luz, agua, etcétera.

El centro de datos tiene capacidad para alojar 15.000 servidores, lo que significa que nuestro servidor es responsable del (suponiendo que un servidor virtual es lo mismo que uno físico, en la práctica suele ser 1 a 16, aproximadamente) un 0,0066% del impacto medioambiental total del centro de datos. El consumo es de 92W por servidor de estas características, lo que supone que el centro de datos consume en total 1.380.000 W de potencia, o lo que es lo mismo, 1.380 kW en servidores.

El consumo total de potencia, atendiendo al PUE que facilita la compañía, se estima en 1.840 kW. Puesto que estos servidores están encendidos continuamente, las 24 horas del día, los 365 días del año, podemos obtener la energía consumida en un año con la siguiente fórmula (Fórmula 6.1):

$$E = \int_{t_0}^{t_f} P\,(t) \cdot dt = P_{cte} \cdot \Delta t = \frac{1.840}{15.000} \cdot n_{serv} \cdot 24 \cdot 365 = n_{serv} \cdot 1.074{,}56\ [kWh] \quad (6.1)$$

Donde n es el número de servidores que se posean.

En nuestro caso n vale 1, así que el servidor consume (contando el coste distribuido de los servicios auxiliares de climatización e infraestructuras unos 1080 kWh anuales.

Esto, equivale aproximadamente, si se alimenta con la red eléctrica nacional, peninsular y convencional (mezcla de energías procedentes de fuentes renovables y no renovables) a 402 kg de CO_2 anuales[9] (Ministerio de Industria Energía y Turismo, 2014).

[9] Se ha utilizado el factor proporcionado por el Ministerio de Industria, Energía y Turismo que vale 0,372.

6.3 PLANIFICACIÓN Y PROGRAMACIÓN DEL TRABAJO FUTURO PROPUESTO

6.3.1 Implementación utilizando *websockets*

Una de las funcionalidades que podrían añadirse, y quizás la más importante, es modificar el sistema para utilizar una solución de "*websockets*" para suministrar los datos. Esto proporcionaría una conexión más directa, que pasaría por memoria RAM, y no por un disco duro, como es el caso de la base de datos SQL. Es cierto que se puede configurar SQL para utilizar la RAM, pero si queremos hacer que funcione a mayores velocidades, debemos forzosamente utilizar "websockets".

En este proyecto no se han utilizado "*websockets"* porque aumentaba considerablemente la dificultad de diseño y la mejora se producía en un rango de frecuencias de actualización que escapaban de los objetivos del mismo. Esto puede realizarlo una persona en tres semanas.

6.3.2 Funcionalidades adicionales

Funcionalidades adicionales como alertas, mejoras en la interfaz, mayor redundancia o la inclusión de herramientas de análisis de datos en la plataforma mejorarían considerablemente su utilidad.

Las alertas, podrían configurarse de modo que cuando algún parámetro se saliera de los valores preconfigurados, el usuario recibiera una alerta en su correo electrónico o por SMS. Esta tarea se puede realizar por una persona en 4 días (trabajando 4 horas cada día).

Una mayor redundancia, significaría realizar una segunda revisión de todos los textos que se imprimen en la pantalla, y crear más hipervínculos o enlaces a los contenidos referidos por los textos. Esta tarea podría realizarse en 2 día, una vez implementadas las anteriores.

La inclusión de herramientas para el tratamiento online de los datos sería de utilidad para, por ejemplo, realizar análisis de Fourier sobre las grabaciones registradas en el sistema. De momento la única manera de lograr esto es descargando los datos en formato Excel® y darles un tratamiento informático mediante este programa. Esta tarea podría realizarse en 4 semanas, a razón de 4 horas diarias de trabajo.

6.3.3 Nota sobre escalabilidad

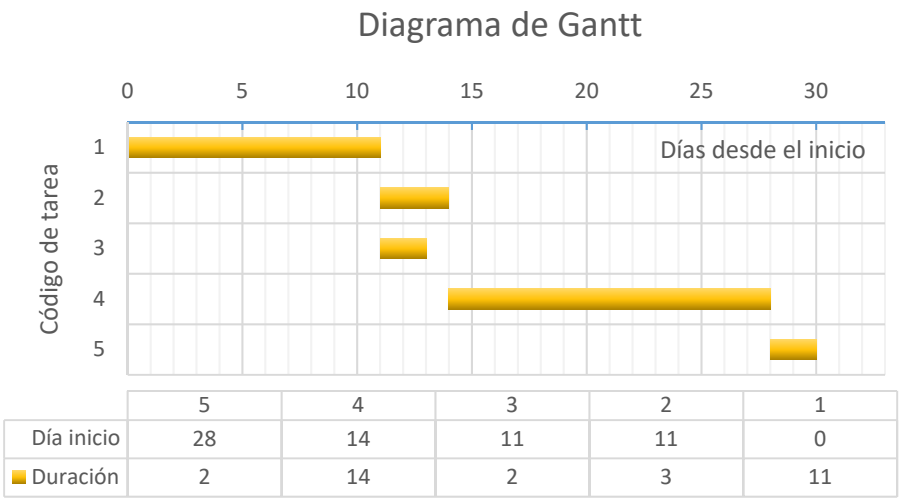
El sistema es completamente escalable: en caso de necesidad sólo hay que mover el programa a un servidor con mayores recursos, un sistema operativo más eficiente, utilizar el compilador JIT (Just in time) o "justo a tiempo" HVMM[10] en lugar del intérprete de PHP (Zend Engine 3) para ejecutar el código, o utilizar "mirrors" (servidores de replicación) distribuidos en posiciones geográficamente próximas al usuario final para disminuir los tiempos de respuesta.

[10] HVMM es una tecnología desarrollada por Facebook™ que compila el código PHP, en lugar de interpretarlo.

6.3.4 Programación temporal

En el apartado anterior (6.3) se han definido las tareas futuras a implementar y el tiempo que podría llevar cada una. No obstante, se ha considerado incluir un diagrama de Gantt porque hay tareas que pueden realizarse paralelamente, trabajando en equipo, y el tiempo total no equivale entonces a la suma de los tiempos individuales de cada tarea.

La siguiente figura (Figura 6.3) muestra un diagrama de Gantt con la planificación de estas tareas teniendo en cuenta un equipo de dos personas y una carga de trabajo de cuatro horas diarias por persona:



Fuente: Elaboración propia

Figura 6.3: Diagrama de Gantt del trabajo futuro propuesto.

Cabe destacar que en todas las tareas hay dos personas trabajando excepto en la tarea número dos, que los dos primeros días hay una (la otra está ocupada en la tarea número tres) y cuando queda libre se suma a la dos, de forma que, de cuatro días para una persona, se invierten tres solamente. Estos detalles y los códigos de tarea pueden observarse mejor en la siguiente tabla (Tabla 6.4):

Tabla 6.4: Planificación del trabajo futuro propuesto.

Tarea	Cód. Tarea	Día inicial	Duración	Día final	Recursos	Requisitos
Websockets	1	0	11	11	2	-
Alertas	2	11	3	14	1p/2d+2p/1d[11]	1
Redundancia	3	11	2	13	1	1, 2
Herramientas	4	14	14	28	2	1
Depuración	5	28	2	30	2	1,2,3 y 4

FUENTE: Elaboración propia.

[11] Una persona dos días y dos personas un día.

6.4 RESULTADOS LOGRADOS: TRANSPORTE DE SEÑALES

6.4.1 Pertinencia, objetivo y definición general del experimento

La fase de implementación de la plataforma ya incluye valoración de resultados, siendo sumamente difícil separar conceptualmente la implementación del análisis de resultados. Dada la naturaleza del proyecto los resultados se han ido mostrando y analizando durante todo el informe, pues la creación de la plataforma en sí ya es un gran resultado por sí mismo. No obstante, y habiendo hecho este comentario, sí que es claramente un resultado final analizar qué tipo de señales y de qué características es capaz de admitir la plataforma y qué distorsiones sufren las mismas. A pesar de estar optimizada para almacenar variables digitales puntuales, es decir, que no formen parte de una "onda" o curva analógica, la plataforma las admite también. Es de interés ver qué tipo admite y con qué fidelidad.

Para ello se probará una señal triangular y una cuadrada. Cada una tiene su propósito: la triangular medir periodos y la cuadrada observar los tiempos de flanco de subida y bajada; así como aberraciones o deformaciones de cualquier índole y medir el tiempo de respuesta que la plataforma DroneLink™ es capaz de alcanzar.

Se pondrán las características de la señal empleada en cada momento y se comparará con una señal visualizada directamente del puerto serie (sin enviarse a internet) utilizando una herramienta de depuración que incluye la plataforma DroneLink™: el software serial plotter (descrito en el apdo. 5.4.3.3). La conexión a internet (velocidad y fallos de transmisión) se simula con las herramientas que proporciona Google® Chrome™.

6.4.2 Diseño técnico del experimento

La transmisión de datos se realizará utilizando la tecnología Bluetooth™ por conveniencia. Se ha hecho una prueba previa muy sencilla consistente en mandar las mismas señales aleatorias utilizando Bluetooth™, XBee™ y cable USB y se ha determinado que todas éstas son capaces de transmitir a 9600 baudios en todos los casos sin ningún problema. No procede pues repetir el experimento para las tres tecnologías principales de transmisión de datos.

Para caracterizar las señales enviadas y recibidas y establecer criterios estables que permitan su posterior comparación es necesario definir correctamente los parámetros a medir y las metodologías empleadas.

En nuestro caso estamos interesados en evaluar posibles diferencias en: la pendiente de las señales, las frecuencias (de entrada y salida), la forma de las señales y el rango de valores de ambas señales. Para ello se han diseñado y se definen una serie de pruebas que se describen en los siguientes subapartados.

La velocidad de conexión será para todos los casos simulada a una velocidad de conexión 3G de cobertura regular. Este es el caso más desfavorable, pero probable, hoy en día en caso de que se

consulte y/o actualicen los datos utilizando un dispositivo móvil actuando de enrutador bajo cobertura 3G, puesto que la 4G no siempre está disponible. Las señales transmitidas son de 1Hz, frecuencia considerada más que suficiente para la razón de ser de la plataforma (no está diseñada para transmitir audio, por ejemplo).

6.4.2.1 Prueba de la pendiente de semiperiodo

Se trata de tomar mediciones (equidistantes y repartidas por ambos semiperiodos de la señal) de las pendientes de las señales de entrada y salida calculando sus derivadas en todos los puntos de las mismas. El intervalo será, como máximo, de 20 ms. Después se hace la media para ambas señales obteniendo cuatro valores (ambos por señal) y el criterio de comparación de las medias será el de igualdad. Cuanto más se parezcan entre sí los valores dos a dos, más similares son las señales. Lo de los semiperiodos es necesario porque de lo contrario, al tratarse de señales periódicas, la media daría próxima a 0.

6.4.2.2 Prueba de frecuencia

Se trata de medir la frecuencia de las señales y comparar ambos valores. En caso de ser una señal no periódica se ponderará el espectro de Fourier con las amplitudes en escala decimal (no logarítmica) y se obtendrá un solo valor de frecuencia representativa o $\overline{f_{rep}}$. Nuevamente se aplica el criterio de igualdad: a mayor similitud de valores, mayor similitud de señales.

6.4.2.3 Prueba de similitud visual

Esta es una prueba cualitativa en la que se incluirán representaciones gráficas de ambas frecuencias de modo que pueda ser contrastada fácilmente. Se establecerá una calificación en una escala de cinco valores que son, de mejor a peor calidad: MS (muy similar), S (similar), R (regular), D (distinta), I (incomparables, señales distintas).

6.4.2.4 Prueba de rango: mínimo y máximo

Se trata de extraer el máximo y el mínimo del recorrido de las señales y compararlos. Deberían de ser idealmente iguales.

6.4.2.5 Prueba de media y varianza

Esta prueba consiste en extraer la media y la varianza de los valores muestrales discretizados a una resolución significativa (5 veces el criterio de Nyquist-Shannon) de ambas señales. Luego se comparan ambos valores. Idealmente la media valdrá 0 si la señal es periódica.

6.4.2.6 Criterio general y tablas resumen

En todos los casos de comparaciones cuantitativas se extraerá una calificación del 0 al 5, ambos incluidos. Si la prueba no puede realizarse por problemas de dominio se asignará S/C y no computará

a efectos de la media final. La puntuación final se indica con un número de estrellas siendo cinco la mayor fidelidad de transmisión y 0 la peor. La resolución de la escala será de 1, excepto la de la media total, o puntuación final, que será de 0,5 redondeando a la media unidad más próxima por arriba.

Esta puntuación final será la media de las puntuaciones individuales en esta misma escala de cada una de las cinco pruebas.

Se restarán los dos valores y este error absoluto se convertirá a relativo tomando como referencia (dividiendo por) el valor de la señal considerada buena o fidedigna: la señal original. Para las pruebas con dos resultados se asignarán puntuaciones individuales a ambos y la total de la prueba será la media aritmética de ambos valores. Se redondeará siempre al menor valor para obtener la puntuación en estrellas.

Por ejemplo, un error relativo de 9% se interpretará como calidad "Buena", y no "Regular". El error relativo se expresará con una cifra significativa. La siguiente tabla (Tabla 6.5) muestra cómo se asignarán las puntuaciones en cada uno de los apartados:

Tabla 6.5: Método de calificaciones individuales de las pruebas realizadas.

Calidad de la señal	Decimales (redondeo)	Error relativo	Puntuación
Muy Buena	2	0%	★ ★ ★ ★ ★
Buena	2	± 5,0%	★ ★ ★ ★
Regular	2	± 10,0%	★ ★ ★
Mala	2	± 20,0%	★ ★
Muy Mala	2	± 30,0%	★
Nefasta	2	± 50,0%	

FUENTE: Elaboración propia.

Para las pruebas cuantitativas el criterio será traducir directamente y en orden de mejor calidad a peor la calificación cualitativa a puntuación en estrellas. No se podrán asignar, por tanto, calificaciones intermedias en esta escala ni valores de 0 estrellas.

La siguiente tabla (Tabla 6.6) recoge y sintetiza las distintas pruebas desarrolladas y llevadas a cabo en este experimento:

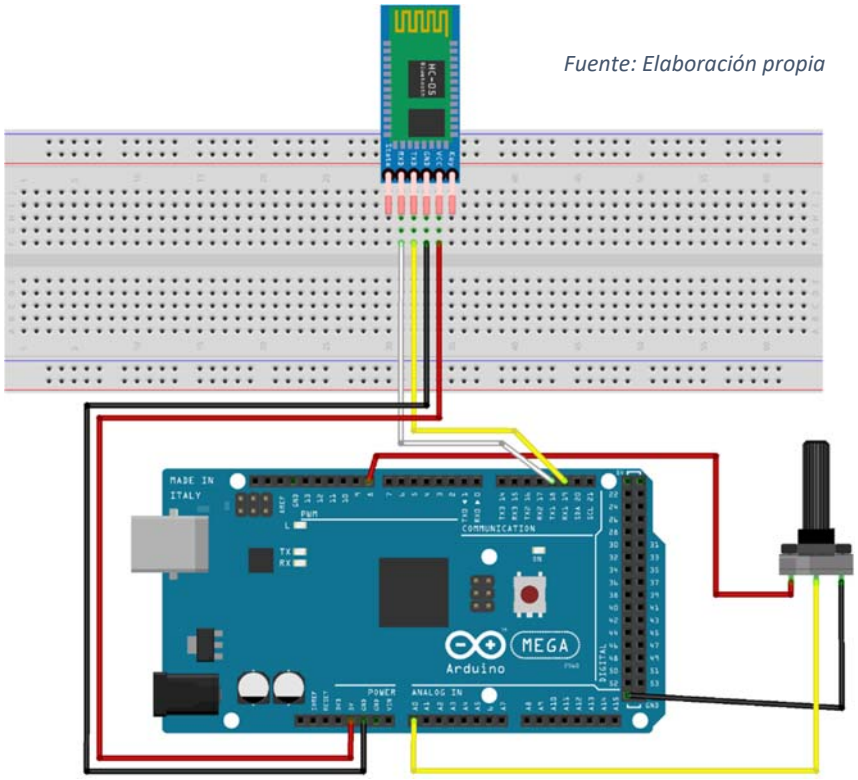
Tabla 6.6: Resumen de pruebas realizadas.

Prueba	Código	Tipo de prueba	Criterio
Pendiente de semiperiodo	A	Cuantitativa	Igualdad
Frecuencia	B	Cuantitativa	Igualdad
Similitud visual	C	Cualitativa	Cualitativo
De rango: mínimo y máximo	D	Cuantitativa	Igualdad
Prueba de media y varianza	E	Cuantitativa	Igualdad

FUENTE: Elaboración propia

6.4.3 Montaje realizado

El dispositivo utilizado para la prueba es un Arduino™ Mega 2560 conectado a un módulo Bluetooth™ HC-05 y alimentado por USB a +5 voltios. Es importante resaltar que el cable USB solamente se utiliza para la alimentación eléctrica del Arduino™ y para transmitir la señal de referencia considerada exacta. Las siguientes figuras muestran el montaje experimental empleado (Figura 6.4 y Figura 6.5):



Fuente: Elaboración propia

Figura 6.4: Montaje realizado para medir la calidad de transmisión de las señales analógicas.

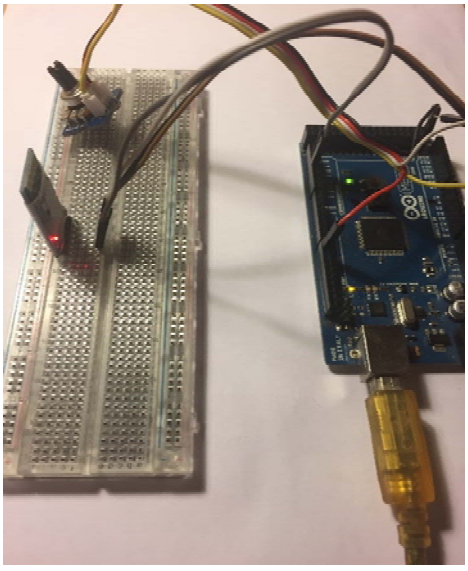


Figura 6.5: Fotografía del montaje realizado.
Fuente: Elaboración propia

6.4.4 Resultados del experimento

Se presentará una imagen con la representación gráfica de la señal depurada por puerto serie (siempre en el formato de la plataforma DroneLink™, especificado en el capítulo 5.4.2.1.1, utilizando el *software* "serial plotter" desarrollado a tales efectos). Al lado se comparará con una imagen del gráfico generado a partir de los datos descargados de la plataforma DroneLink™.

6.4.4.1 Señal triangular

Para el apartado de la señal triangular, al ser una señal periódica, y dada la naturaleza de esta, se observa como los dos valores de la prueba A, o la de frecuencia de semiperiodo son opuestos. La media de ambos semiperiodos tiene el mismo valor absoluto, razón por la que se incluye el símbolo de "más menos" delante del resultado, para que se vea que hay dos posibles valores en función de si es el semiperiodo de subida o el de bajada.

Es un buen signo que esta propiedad se mantenga en la señal procesada, pues indica que no hay diferencias en función del semiperiodo en el que nos encontramos. También es indicativo de que, con alta probabilidad, no existen deformaciones de la señal que sean función del tiempo (podría ser que existiera una deformación que fuera función periódica, también, del tiempo y que los periodos fueran múltiplos del periodo de la señal y que, por tanto, no pudiera apreciarse bien la falta de errores con esta prueba, pero sería un caso aislado muy poco probable).

Los resultados obtenidos se muestran a continuación en la siguiente figura (Figura 6.6) y en la siguiente tabla (Tabla 6.7):

Fuente: Elaboración propia

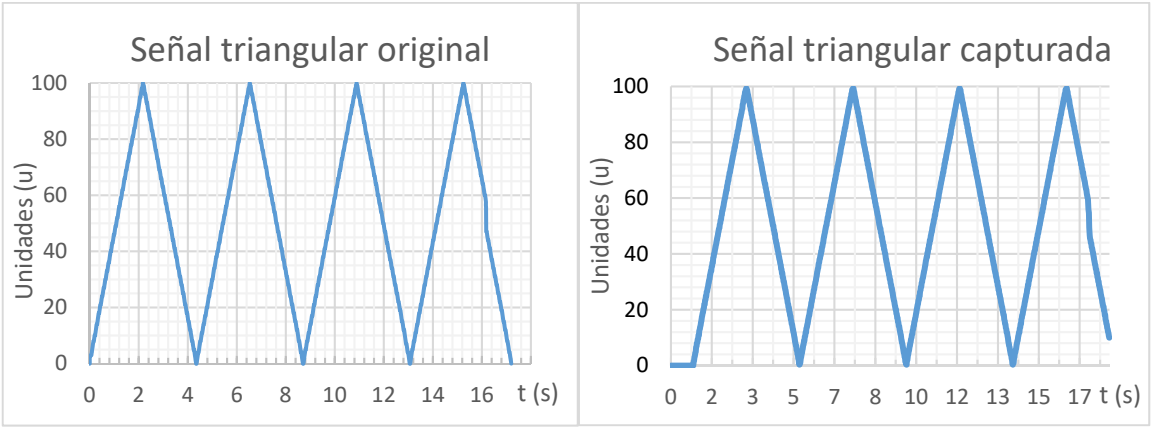


Figura 6.6: Comparación entre la señal triangular enviada y la recibida.

Tabla 6.7: Resultados del experimento con onda triangular.

Ref.	Valor(es) I	Valor(es) II	Error Abs.	Error Rel.	Cal. Cuant.	Calificación prueba
A	±60,24 u/s	±58,72 u/s	1,46 u/s	±2,42 %	Muy Buena	★ ★ ★ ★ ★
B	1,14 Hz	1,22 Hz	0,08 Hz	7,02 %	Buena	★ ★ ★ ★
C	-	-	-	-	S (Similar)	★ ★ ★ ★
D	0,00; 100,00 u	0,00; 100,0 u	0,00 u	0,00 %	Muy Buena	★ ★ ★ ★ ★
E	47,23 u; 885,17 u^2	46,83 u; 884,52 u^2	0,40 u; 0,65 u	0,84 %; 0,07 %	Muy Buena	★ ★ ★ ★ ★

FUENTE: Elaboración propia.

Cabe destacar que las características de la señal enviada se corresponden con lo que podría ser un flujo de datos para el que la plataforma fue diseñada: de aproximadamente una operación (actualización o lectura) autenticada por segundo. Si se eliminasen los mecanismos de seguridad, la velocidad de procesamiento aumentaría notablemente, pero, por contrapartida, los datos quedarían expuestos a cualquier persona.

La calificación de la prueba es de 4,6, que al convertirse a calificación global según los procedimientos anteriormente expuestos resulta en un total de 5 estrellas.

6.4.4.2 Señal cuadrada

La siguiente figura (Figura 6.7) muestra la señal cuadrada de 1 Hz enviada a la plataforma DroneLink™ desde el modelo de pruebas:

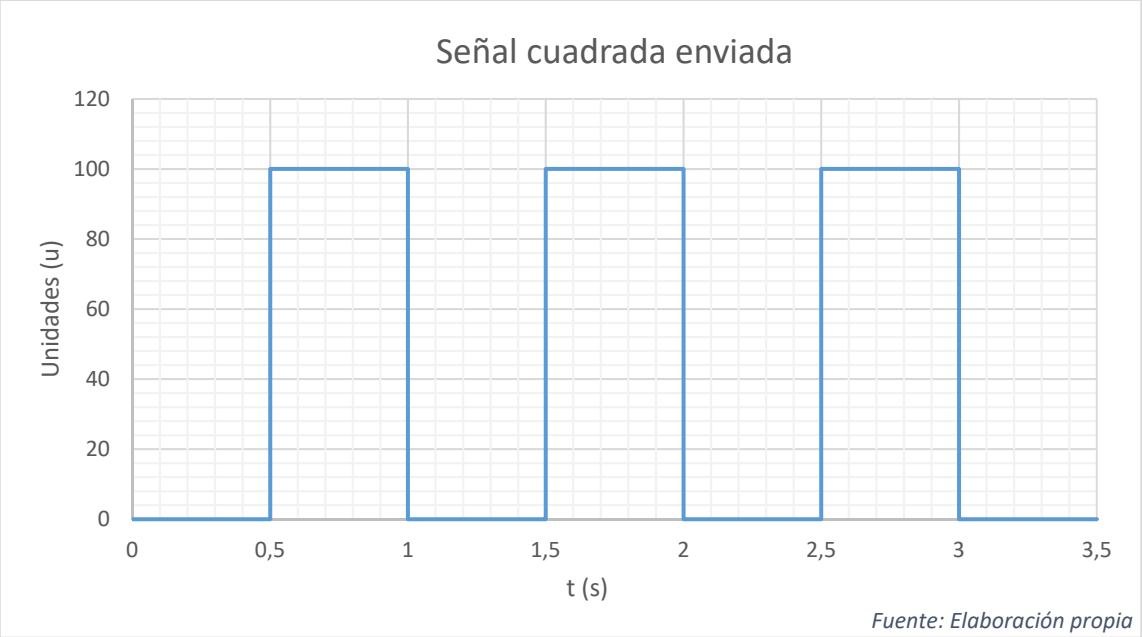
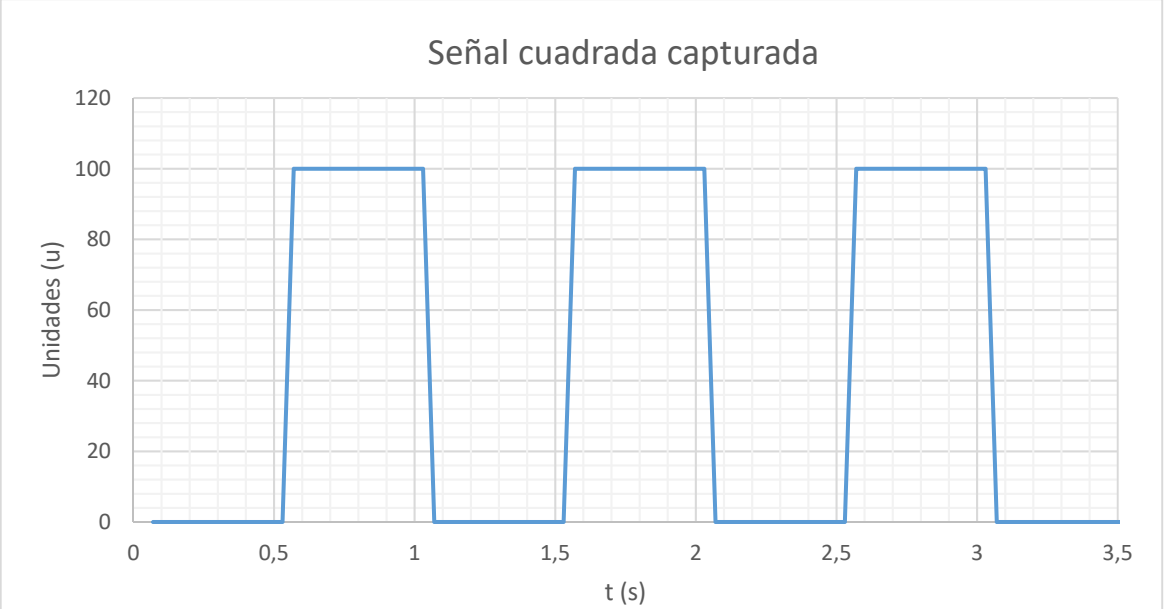


Figura 6.7: Señal cuadrada de 1 Hz enviada a la plataforma DroneLink™

La señal recibida en la plataforma se ve tal y como muestra el siguiente gráfico (Figura 6.8):



Fuente: Elaboración propia

Figura 6.8: Señal cuadrada capturada por la plataforma DroneLink™

Se aprecian ligeros flancos de subida y bajada debido a retardos en la trasmisión. Este era una de las principales motivaciones para probar una onda cuadrada. La pendiente de flanco o *slew* rate, en inglés, es de 1000 u/s. Los resultados del sistema de calificación por pruebas son los que muestra la tabla que aparece a continuación (Tabla 6.8):

Tabla 6.8: Resultados del experimento con onda cuadrada.

Ref.	Valor(es) I	Valor(es) II	Error Abs.	Error Rel.	Cal. Cuant.	Calificación prueba
A	-	-	-	-	-	S/C
B	1 Hz	0,98 Hz	0,02 Hz	2,00 %	Muy Buena	★ ★ ★ ★ ★
C	-	-	-	-	S (Similar)	★ ★ ★ ★
D	0,00; 100,00 u	0,00; 100,0 u	0,00 u	0,00 %	Muy Buena	★ ★ ★ ★ ★
E	42,86 u; 2637,36 u^2	42,86 u; 2637,36 u^2	0,00 u; 0,00 u^2	0,00 %; 0,00 %	Muy Buena	★ ★ ★ ★ ★

FUENTE: Elaboración propia.

El promedio total de la señal cuadrada es de 4,75, que, aplicando los criterios de redondeo, se convierte en una calificación de 5 estrellas.

6.4.4.3 Puntuación total de la plataforma

La puntuación total de la plataforma es la puntuación media de las dos ondas (triangular y cuadrada), resultando en 5 estrellas (★★★★★). La plataforma alcanza, pues, la puntuación máxima del test para transmisiones próximas al hercio. Cabe recordar al lector que esta plataforma no fue diseñada para la transmisión de señales originalmente, sino para la transmisión de valores discretos puntuales en tiempo real.

6.5 CONCLUSIONES Y RECOMENDACIONES DE CONTINUACIÓN DEL PROYECTO

El proyecto concluye con la creación de una plataforma que permite la comunicación segura y confiable entre distintos dispositivos (especialmente drones) en tiempo real. Este era el objetivo principal, y se puede, por tanto, considerar cumplido con un total de 4265 líneas de código escritas.

La creación de librerías, programas o utilidades para conectar al sistema la mayor parte de dispositivos inteligentes que existen en el mercado actualmente también es algo que se ha cumplido, como ha sido extensamente explicado a lo largo y ancho de este informe.

La seguridad, que es otra de las preocupaciones del trabajo, se ha conseguido implementar a nivel de código y a nivel de transporte de datos utilizando protocolos de comunicación seguros.

Como recomendación de continuación del proyecto sugiero, tal y como se ha comentado en el apartado anterior, intentar implementar de nuevo el mismo, pero tratando de utilizar tecnologías de conexión directas: sin pasar por un servidor de base de datos. Después de esto, llevar a cabo la serie de medidas adicionales comentadas en el apartado 6.3.2.

Una vez implementada del todo la comunicación bidireccional del sistema, sería ventajoso incluir en la plataforma herramientas que se aprovechen de algoritmos de control automático para controlar drones individuales, o enjambres de éstos, aprovechando la capacidad de cómputo de un ordenador grande, potente y externo al mismo que sería imposible cargar encima del dron. De este modo se podrían integrar mecanismos de coordinación entre los distintos drones y, en un futuro, como se ha explicado, llegar incluso a realizar el procesamiento de las actuaciones en tiempo real en la plataforma y mandar al dron la información ya procesada.

Una de las principales dificultades encontradas durante la realización de este proyecto ha sido la complejidad que supone recabar información sobre los mecanismos de comunicación que utilizan los drones, puesto que cada fabricante utiliza sus protocolos particulares y no suele proporcionar información al respecto.

Con todo, la parte de la construcción del modelo de pruebas o prototipo, fue una de las partes de este proyecto más sencillas y que mejor funcionó: al primer intento.

Finalmente, como en todo proyecto técnico, se ha encontrado la dificultad típica de este tipo de proyectos y que es inherente a la implementación de cualquier tecnología: el salto del papel a la práctica. Pequeñas dificultades, ajustes y quebraderos de cabeza que impedían que tal valor fuera actualizado o que tal conexión se realizara correctamente. Este tipo de problemas han surgido constantemente, sobretodo en la fase de programación, donde cada vez que se introducía una función nueva se perdía una cantidad ingente de tiempo solucionando errores después de programada.

6.6 REFERENCIAS BIBLIOGRÁFICAS Y NORMATIVA APLICADA

Arsys. (2012). Arsys mejora hasta 1,35 el PUE de su Centro de Datos, 1. Retrieved from http://conoce.arsys.es/arsys-mejora-hasta-135-el-pue-de-su-centro-de-datos/

Bicsi, B. (2002). *Network Design Basics for Cabling Professionals* (1ª). McGraw Hill Professional.

Bluetooth Core Specification v3.0 + HS. (2009). Retrieved from http://www.bluetooth.com/Bluetooth/Technology/Building/Specifications

BTSIG. (2006). Simple Pairing Whitepaper. *Bluetooth Special Interest Group*, *1*(1), 23. Retrieved from http://web.archive.org/web/20061018032605/http://www.bluetooth.com/NR/rdonlyres/0A0B3F36-D15F-4470-85A6-F2CCFA26F70F/0/SimplePairing_WP_V10r00.pdf

Cisco Systems IoT. (2016). Retrieved from http://www.cisco.com/c/en/us/solutions/internet-of-things/overview.html

Coppley, J. (2015). Diversifiying the IoT with Sub-1 GHz technology. *Texas Instruments Publication*. Retrieved from http://www.ti.com/lit/wp/swry017/swry017.pdf

Cutting edge: IT's guide to edge data centers. (2016). Retrieved December 17, 2016, from http://internetofthingsagenda.techtarget.com/definition/Internet-of-Things-IoT

Eco, U. (2009). *COMO SE HACE UNA TESIS: TECNICAS Y PROCEDIMIENTOS DE ESTUDIO, INV ESTIGACION Y ESCRITURA (6ª ED.)*. (L. Baranda & A. Clavería, Eds.). Madrid: Gedisa.

El globo aerostático y la conquista de los cielos. (2014). Retrieved September 27, 2016, from http://www.nationalgeographic.com.es/historia/grandes-reportajes/el-globo-aerostatico-y-la-conquista-de-los-cielos_7848

El uso adecuado del prefacio. (2009). Retrieved from http://www.ehowenespanol.com/adecuado-del-prefacio-info_355550/

Esteso, M. P. (2014). Tutorial de Arduino: comenzando con XBEE. Retrieved November 12, 2016, from https://geekytheory.com/tutorial-arduino-comenzando-con-xbee

Geffrey, C. (2009). The History of Network Cabling. *Data Cottage*. misc.

González García, Juana Mª; León Mejía, A. P. S. M. (2012). *Cómo escribir un trabajo de fin de grado* (1st ed.). Madrid: Síntesis S.A.

Hernándo, J. M. (1991). *Sistemas de telecomunicación* (2ª). Servicio de publicaciones E.T.S.I. Telecomunicación.

Historia de Ethernet. (2014). Retrieved November 21, 2016, from http://www.alfinal.com/Temas/ethernet.php

Holmes, R. (2012). *La edad de los prodigios: terror y belleza en la ciencia del Romanticismo.* (1ª). Madrid: Turner.

International Telecommunication Union. (1994). X.200: Data Networks and open system communications, *4*, 59. Retrieved from https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-X.200-199407-I!!PDF-E&type=items

Kerry, T. (2014). Exploring XBEE and XCTU. Retrieved December 7, 2016, from https://learn.sparkfun.com/tutorials/exploring-xbees-and-xctu

Kharif, O. (2014). Cisco's CEO Pegs internet of things. Retrieved October 27, 2016, from

https://www.bloomberg.com/news/articles/2014-01-08/cisco-ceo-pegs-internet-of-things-as-19-trillion-market

Lesurf, J. (2004). Transistor-Transistor Logic.

MCI Electronics. (2012). XBee Tutorial. Retrieved December 14, 2016, from http://xbee.cl/tutorial-xbee/

Mehdi, A., Betz, F., Richter, G., Seguy, D., & Vrana, J. (2016). PHP Reference (Manual). Retrieved September 25, 2016, from http://php.net/manual/es/index.php

Ministerio de Industria Energía y Turismo. (2014). *Factores de emisión de CO2 y coeficientes de paso a energía primaria a partir de diferentes fuentes de energía final consumidas en el sector edificios en España*. Madrid.

Puerto serie y puerto paralelo. (2016). Retrieved September 17, 2016, from http://es.ccm.net/contents/404-puerto-serial-y-puerto-paralelo

Qué es un XBEE y por qué es necesario. (2014). Retrieved October 27, 2016, from https://mecatronicauaslp.wordpress.com/2013/07/04/xbee-parte-1-que-es-un-xbee-y-que-es-necesario/

Sean, A. (2016). C++ Tutorial and Guide. Retrieved October 21, 2016, from http://www.cplusplus.com/

SQL Tutorial. (2010). Retrieved September 26, 2016, from http://www.w3schools.com/sql/

Talmor, L. (n.d.). Where to Begin? When, Where and How to Write a Prologue. Retrieved November 28, 2016, from http://www.writing-world.com/fiction/prologue.shtml

Zimmermann, H. (1980). OSI Reference Model-The ISO Model of Architecture for Open Systems Interconnection. *IEEE Transactions on Communications*, *28*(4), 425–432. http://doi.org/10.1109/TCOM.1980.1094702

Se ha aplicado la normativa de citación bibliográfica de la Asociación Americana de Psicología (APA, por sus siglas en inglés), en su 6ª edición.

7 ÍNDICE ANALÍTICO

P

S

T

U

X